“UNIVERSIDAD NACIONAL DE PIURA”

FACULTAD DE CIENCIAS

ESCUELA PROFESIONAL DE INGENIERÍA ELECTRÓNICA Y TELECOMUNICACIONES

“DISEÑO DE UN SISTEMA AUTOMATIZADO PARA CONTROL Y SUPERVISIÓN DE UN ACUARIO USANDO TECNOLOGÍAS INALÁMBRICAS GPRS Y BLUETOOTH”.

PRESENTADA POR:

BACH. LUIS ARTURO GARCIA NORIEGA

TESIS PARA OPTAR EL TÍTULO PROFESIONAL DE INGENIERO ELECTRÓNICO Y TELECOMUNICACIONES

LINEA INVESTIGACIÓN: INFORMATICA ELECTRÓNICA Y TELECOMUNICACIONES

SUBLINEA DE INVESTIGACIÓN: SISTEMA DIGITAL

PIURA – PERÚ

2018

TESIS PRESENTADA COMO REQUISITO PARA OPTAR EL TÍTULO DE INGENIERO ELECTRÓNICO Y TELECOMUNICACIONES

ASESOR:

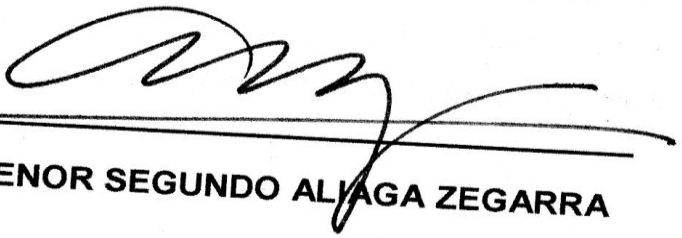
ING. FRANKLIN BARRA ZAPATA

TESISTAS:

LUIS ARTURO GARCIA NORIEGA

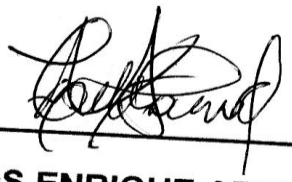
JURADO EVALUADOR:

PRESIDENTE:



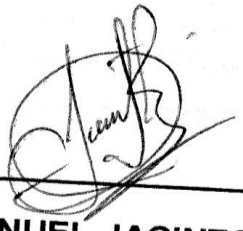
DR. ANTENOR SEGUNDO ALIAGA ZEGARRA

SECRETARIO:



DR. CARLOS ENRIQUE ARELLANO RAMIREZ

VOCAL:



MSC. JUAN MANUEL JACINTO SANDOVAL

TESIS PRESENTADA COMO REQUISITO PARA OPTAR EL TÍTULO DE INGENIERO ELECTRÓNICO Y TELECOMUNICACIONES

DECLARACION JURADA DE AUTENTICIDAD DE TESIS

Yo, **LUIS ARTURO GARCIA NORIEGA**, identificado con DNI N° 45737952 bachiller de la Escuela Profesional de Ingeniería Electrónica y Telecomunicaciones de la Facultad de Ciencias de la Universidad Nacional de Piura con Domicilio MZ. A LOTE 23 CESAR VALLEJO DISTRITO 26 DE OCTUBRE – PIURA con celular número 994538802, correo Electrónico lagnoriega@gmail.com ante usted con el debido respeto me presento y expongo:

DECLARO BAJO JURAMENTO:

Que la tesis que presento es auténtica no siendo copia parcial y total de una tesis desarrollada y/o realizada en el Perú o el extranjero en caso contrario de encontrar falsa la información que proporciono me sujeto a los alcances de lo establecido en el art. N°411 del código penal concordante con el art. N° 27444, y ley del procedimiento administrativo general y las normas legales de protección a los derechos de autor.

En la Fe de lo cual firmo la presente

Piura 06 de Septiembre del 2018

LUIS ARTURO GARCIA NORIEGA

DNI: 45737952

UNIVERSIDAD NACIONAL DE PIURA
FACULTAD DE CIENCIAS

ACTA DE SUSTENTACIÓN 048-2018-D-FC-UNP

Los Miembros del Jurado Calificador que suscriben, reunidos para evaluar la Tesis denominada **"DISEÑO DE UN SISTEMA AUTOMATIZADO PARA CONTROL Y SUPERVISIÓN DE UN ACUARIO USANDO TECNOLOGÍAS INALÁMBRICAS GPRS Y BLUETOOH"** presentado por el señor Bachiller **GARCÍA NORIEGA – LUIS ARTURO,** con el asesoramiento del **M.Sc. FRANKLIN BARRA ZAPATA**; oídas las observaciones y respuestas a las preguntas formuladas, y de conformidad al Reglamento de Tesis para obtener el Título Profesional en la Facultad de Ciencias, lo declaran:

APROBADO (X) **DESAPROBADO ( )**

Con la mención de:

Muy Bueno

(X) En consecuencia, queda en condición de ser ratificado por el Consejo de Facultad de Ciencias de la Universidad Nacional de Piura, y recibir el **TITULO PROFESIONAL DE INGENIERO ELECTRÓNICO Y TELECOMUNICACIONES.**

(X) En consecuencia, queda en condición de ser ratificado por el Consejo Universitario de la Universidad Nacional de Piura, y recibir el **TITULO PROFESIONAL DE INGENIERO ELECTRÓNICO Y TELECOMUNICACIONES**; después que el sustentante incorpore la sugerencia del Jurado Calificador.

Piura, 14 de agosto 2018.

DR. ANTENOR SEGUNDO ALIAGA ZEGARRA
PRESIDENTE DE JURADO DE TESIS

DR. CARLOS ENRIQUE ARELLANO RAMÍREZ
SECRETARIO DE JURADO DE TESIS

DEDICATORIA

Este proyecto de Tesis se lo dedico enteramente a mis padres Edmundo y Carmen, puesto que no sería nada si no fuera por ellos, que día a día me apoyaron incondicionalmente en este camino, logrando darme ánimos en momentos en los cuales podía haberme desmoronado, haciéndome ver mis habilidades, fortaleciendo esos puntos en donde me debilitaba y dándome ese gran amor que hasta ahora sigue siendo el motor que me impulsa a superarme día a día… Gracias infinitas a uds papitos, les amo y este paso lo di por ustedes

LUIS ARTURO GARCIA NORIEGA

AGRADECIMIENTO

Esta tesis es el resultado de la constancia y sacrificio, del apoyo reunido de aquellas personas que decidieron aportar en éste logro con su experiencia, tiempo y conocimientos, ya que han desarrollado un papel importante en este material, empezando por mi asesor de tesis el Ing. Franklin Barra Zapata a quién agradezco por su apoyo incondicional, por el tiempo que dispuso para guiarme en mi proyecto **"DISEÑO DE UN SISTEMA AUTOMATIZADO PARA CONTROL Y SUPERVISIÓN DE UN ACUARIO USANDO TECNOLOGÍAS INALÁMBRICAS GPRS Y BLUETOOTH"**

A la comisión de evaluación por sus críticas y comentarios constructivos y acertados durante la elaboración de éste proyecto, por la intervención oportuna para un mejor resultado del presente trabajo.

A cada persona que decidió apoyarme y que hizo posible en ésta parte de mi vida, ayudarme a cumplir un objetivo más, a todos ellos muchas gracias.

LUIS ARTURO GARCIA NORIEGA

CONTENIDO

FIGURAS

RESUMEN

La presente tesis se busca implementar un sistema automatizado para el control y supervisión de un acuario usando tecnologías inalámbricas GPRS y BLUETOOTH. Se diseñará e implementará un módulo de control que recogerá la información de las variables de nivel de agua, temperatura y PH, también se encargará de realizar ciertas tareas periódicas como encender y apagar las luces, filtros bombas de agua y compresores en la pecera y tanque de almacenamiento. El módulo de control de temperatura tomará el valor de sensores de temperatura para manejar al calentador del acuario, el módulo de supervisión del pH deberá permitir monitorizar el valor de pH en el acuario. Cuando las variables monitoreadas de temperatura, nivel y PH superen cierto rango deberá generar una señal de alarma enviando mensajes de texto a un número de celular específico. El sistema deberá permitir, por otro lado, supervisar y controlar el Nivel de agua en el acuario, y realizar los cambios de agua de manera automática al 25% o 50% del agua del acuario. Se implementara un dosificador de alimentos para peces basado en servo motor el cual se le podrá programar cuando y cuantas veces entregara comida en el acuario. Para la comunicación se utilizara módulos Bluetooth esclavo y modem GSM/GPRS SIM900, para lo cual se implementara los respectivos protocolos de comunicación con comandos tipo texto por ejemplo un comando seria 1234V, el cual solicita el estado de la variables monitoreadas.

En el primer capítulo se describe la problemática y se establece el objetivo general y los objetivos específicos de la tesis, así como la justificación, los alcances y los límites de la misma. El segundo capítulo hace referencia al marco teórico y las definiciones que nos permiten entender desde la base la aplicación desarrollada. En el tercer capítulo se muestra el desarrollo del sistema paso a paso tanto en el diseño del hardware como del software. El cuarto capítulo nos permite analizar los costos para la implementación de este proyecto y al final se muestra las conclusiones.

PALABRAS CLAVE: ACUARIOS, PECERAS, ARDUINOMEGA, GSM, SMS, BLUETOOTH, TEMPERATURA, NIVEL, PH.

ABSTRACT

The present thesis seeks to implement an automated system for the control and supervision of an aquarium using wireless technologies GPRS and BLUETOOTH. A control module will be designed and implemented that will collect the information of the variables of water level, temperature and PH, will also be responsible for performing certain periodic tasks such as turning lights on and off, water pump filters and compressors in the tank and tank storage. The temperature control module will take the value of temperature sensors to operate the aquarium heater, the pH monitoring module should allow to monitor the pH value in the aquarium. When the monitored variables of temperature, level and PH exceed a certain range, they must generate an alarm signal by sending text messages to a specific cell number. The system should allow, on the other hand, to monitor and control the water level in the aquarium, and make the water changes automatically to 25% or 50% of the aquarium water. A fish feed doser based on servo motor will be implemented which can be programmed when and how many times to deliver food in the aquarium. For communication, slave Bluetooth modules and GSM / GPRS SIM900 modem will be used, for which the respective communication protocols will be implemented with text commands such as a serious command 1234V, which requests the status of the monitored variables.

In the first chapter the problem is described and the general objective and the specific objectives of the thesis are established, as well as the justification, the scope and the limits of it. The second chapter refers to the theoretical framework and the definitions that allow us to understand the application developed from the base. The third chapter shows the step-by-step development of the system in both hardware and software design. The fourth chapter allows us to analyze the costs for the implementation of this project and in the end the conclusions are shown.

KEYWORDS: AQUARIANS, PECERAS, ARDUINOMEGA, GSM, SMS, BLUETOOTH, TEMPERATURE, LEVEL, PH.

CAPÍTULO 1

PLANTEAMIENTO METODOLÓGICO

1.1. FORMULACIÓN DEL PROBLEMA

1.1.1. DELIMITACION

En concordancia con los requerimientos de un acuario y/o peceras, se estudiaran las condiciones naturales de vida de las diferentes especies de peces que habitan en peceras y que son comunes en la ciudad de Piura, para determinar la temperatura y la concentración de pH que favorezcan a un normal desarrollo de los peces.

Se diseñará e implementará un módulo de control que recogerá la información de las variables de nivel de agua, temperatura y PH.

También se encargará de realizar ciertas tareas periódicas como encender y apagar las luces, filtros bombas de agua y compresores en la pecera y tanque de almacenamiento.

El módulo de control de temperatura tomará el valor de sensores de temperatura para manejar al calentador del acuario.

El módulo de supervisión del pH deberá permitir monitorizar el valor de pH en el acuario. Cuando esta variable se salga de cierto rango deberá generar una señal de alarma.

El sistema deberá permitir, por otro lado, supervisar y controlar el Nivel de agua en el acuario, y realizar los cambios de agua de manera automática al 25% o 50% del agua del acuario.

Se implementara o dosificador de alimento para peces basado en servo motor el cual se le podrá programar cuando y cuantas veces entregara comida en el acuario.

Para la comunicación se utilizara módulos Bluetooth esclavo y modem GSM/GPRS para lo cual se implementara los respectivos protocolos de comunicación.

¿Es posible diseñar un sistema automatizado para control y supervisión de un acuario usando tecnologías inalámbricas GPRS Y BLUETOOTH?

1.2 OBJETIVOS DE LA INVESTIGACIÓN

Diseñar un sistema automatizado para control y supervisión de un acuario usando tecnologías inalámbricas GPRS y BLUETOOTH

1.2.2 OBJETIVOS ESPECÍFICOS

- Determinar los bloques del sistema
- Seleccionar los componentes que tendrá cada bloque del sistema automatizado de control y supervisión
- Monitorear el Nivel, Temperatura y PH en el acuario
- Automatizar el sistema de iluminación, aireación, filtraje entre otros elementos en el acuario.
- Diseñar un sistema de control y supervisión de temperatura en el acuario usando calentadores.
- Diseñar sistema de alarma para la diferentes variables ambientales del acuario
- Implementar Protocolo de comunicaciones para tecnología Bluetooth y GPRS
- Realización de pruebas para la comprobación del funcionamiento del sistema que se propone

1.3. HIPÓTESIS GENERAL

Usando las tecnologías existentes, si es posible diseñar un sistema automatizado para control y supervisión de una pecera en un acuario usando tecnologías inalámbricas GPRS y BLUETOOTH.

1.4. JUSTIFICACION

El proyecto se justifica ya que el sistema automatizado que se propone realizar o desarrollar tiene como finalidad mejorar las condiciones de vida de cada una de las especies de peces que habitan en el acuario.

CAPITULO 2

MARCO TEORICO

2.1 ANTECEDENTES DE LA INVESTIGACIÓN

Según **FÉLIX VICENTE JÍMENEZ RÍOS y JOAQUÍN RIVERO JUÁREZ**, en su Tesis titulada **"DISEÑO Y CONSTRUCCIÓN DE UNA TARJETA PROGRAMABLE DE ADQUISICIÓN, PROCESAMIENTO DE DATOS Y CONTROL"**, indica lo siguiente:

La automatización industrial se divide en 3 fases: adquisición de datos, procesamiento o tratamiento de la información y control de elementos actuadores.

Existen alternativas comerciales que permiten realizar una o todas las fases de automatización en un solo componente como tarjetas de adquisición de datos o controladores lógicos programables (PLCs).

En este trabajo se presenta el diseño, construcción y prueba de una tarjeta programable capaz de realizar las 3 funciones de automatización de un proceso con la capacidad de trabajar de forma autónoma. Se diseñó y fabricó una tarjeta de aplicación para la realización de las pruebas de los módulos de: entradas/salidas digitales, entradas analógicas, potencia para motores, comunicación con la computadora y memorias EEPROM.

Además de las pruebas funcionales, se diseñaron, construyeron y probaron 3 plantas mecatrónicas con la tarjeta programable. Las 3 plantas son: un robot móvil seguidor de línea, un modelo a escala de una plataforma de seguimiento solar y una cámara térmica con temperatura controlada.

Según **Pfarher Iván**, en su artículo titulado **"Tarjeta Entrenadora EA128"**, indica lo siguiente: Este proyecto se basa en una tarjeta de propósitos generales denominada EA128,

su función es el estudio de los microcontroladores AVRs y sus módulos asociados. El objetivo es poder realizar diferentes proyectos y/o prototipos sobre la misma tarjeta, evitando la fabricación de un nuevo hardware. Presenta gran versatilidad en cuanto a la utilización de los periféricos de este microcontrolador, pudiendo setear cada puerto o módulo del mismo sin desperdiciar ningún recurso del hardware.

Con el transcurrir de los años, los microcontroladores comienzan a tener cada vez más importancia en el desarrollo de nuestros circuitos electrónicos, llegando al punto de que cada día nos nace un nuevo proyecto en nuestra mente, el cual queremos hacerlo realidad lo antes posible. Es entonces donde una tarjeta entrenadora de propósito general nos soluciona una buena parte de nuestro problema. Con esto, podemos lograr, asociando un mínimo hardware (en caso de que sea necesario), llevar acabo el prototipo del proyecto y preocuparnos solo en el firmware a desarrollar.

Según **FERNANDO ANDRÉS MORENO PARRA y PAULA ANDREA SEPÚLVEDA HOYOS**, en su Tesis titulada **"SISTEMA DE CONTROL SUPERVISOR DE LAS CONDICIONES AMBIENTALES DE UNA BODEGA DE PECES ORNAMENTALES"**, indica lo siguiente:

A partir del aprovechamiento de la biodiversidad de la cual goza Colombia, ha surgido una forma de empresa nacional dedicada a la comercialización de peces ornamentales. En Bogotá se encuentran bodegas de almacenamiento de peces de este tipo, propiedad de empresas que tienen por objeto la comercialización de peces ornamentales de diferentes variedades con destino al mercado interno y externo del país. Estos peces proceden de la Orinoquía colombiana, Meta, Vichada, Guainía, Amazonas y en menor proporción de la Cienaga en el norte del país, donde cientos de personas se dedican a la actividad de la pesca, capturan variadas especies de peces y los empacan en bolsas de plástico con agua y oxígeno para enviarlos a Bogotá por vía terrestre o aérea; una vez son recibidos en las bodegas, estos son almacenados en acuarios de vidrio clasificados por especie, por un tiempo que puede variar entre 8 días y un mes, mientras son despachados a los clientes.

La situación actual de las bodegas de almacenamiento de peces ornamentales en Bogotá, presenta un escenario caracterizado por la falta de automatización y

control de las variables involucradas en la construcción del ambiente propicio para los peces.

Surge entonces la necesidad de desarrollar un sistema para incrementar la eficiencia de los procesos de control y supervisión de algunas de estas variables, en la etapa de almacenamiento que tiene lugar dentro de la actividad comercial de las organizaciones dedicadas a esta labor, por lo cual el presente proyecto presenta una solución particular a esta situación, mediante el desarrollo de un sistema centralizado encargado de controlar y supervisar de una forma más eficiente, las variables de temperatura, aireación e iluminación.

El centro de este proyecto es la empresa “JB peces tropicales Export”, cuya bodega de almacenamiento posee aproximadamente 500 acuarios, en los cuales se mantienen entre 50 y 100 especies diferentes de peces. El mantenimiento de condiciones como la temperatura y la aireación, son labores realizadas de forma totalmente manual y hasta cierto punto empíricas, no se cuenta con métodos o sistemas de supervisión y control de dichas variables, las tareas se llevan a cabo por ensayo y error o por simple acción repetitiva a partir de la consecución de resultados satisfactorios.

El método empírico que hasta ahora se ha llevado en la bodega presenta fallas, ocasionando alteración en las condiciones ambientales de los peces. Gracias al sistema de control supervisor se tiene un conocimiento inmediato de las alarmas que se presentan cuando ocurran fallas en aireación en los acuarios o cambios en la temperatura, evitando pérdidas económicas representadas en la muerte de cierto número de peces.

Con el desarrollo de éste proyecto no sólo se dio una solución al problema de una compañía en particular, sino que se abrieron las puertas para tecnificar una de las etapas de la comercialización de peces ornamentales como es su almacenamiento.

Este documento presenta el desarrollo del sistema propuesto como solución a la supervisión de la aireación, temperatura e iluminación en la bodega centro de este proyecto. A lo largo de éste se muestra la descripción general del sistema desarrollado, sus especificaciones tanto de hardware como de software, así como los desarrollos que se realizaron para llegar al producto final.

2.2. CARACTERISTICAS DE UN ACUARIO

Puesto que los peces requieren del agua para su existencia, es obvio que el primer paso hacia el acuarismo es adquirir, o construir, el recipiente que ha de contener el líquido en cuestión.

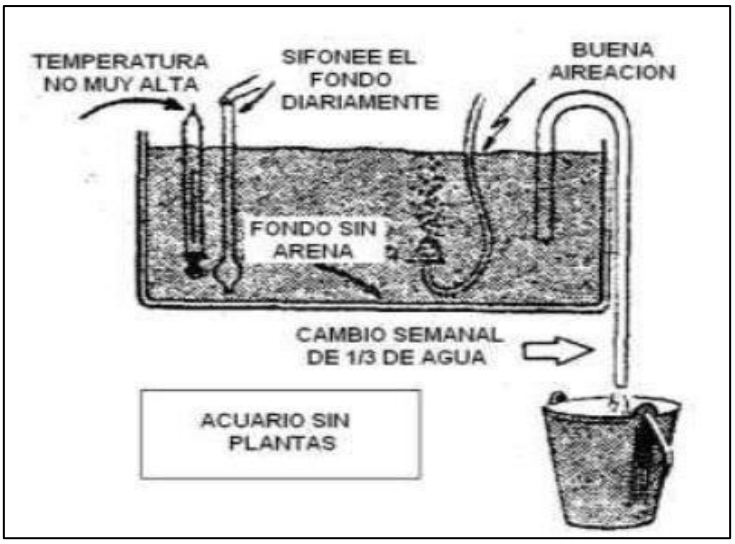
El papel que desempeña el acuario en la decoración moderna es importantísimo, aporta una serie de características destacables, no fácilmente reunidas por otros elementos; a saber: color, luz, atracción, dinámica. Más aún, estas características son tan notables que, en algunos casos, constituye un difícil problema lograr un correcto equilibrio en el ambiente donde se ha efectuado la instalación sin que la mirada se desvíe constantemente hacia el acuario, convertido en espectáculo.

Desde el punto de vista del aficionado acuarista, los aspectos de índole funcional, son los que nos interesan, pero teniendo siempre en cuenta que los problemas estéticos no deben, en modo alguno, perturbar los funcionales, puesto que el acuario constituye, antes que otra cosa, la vivienda ocasional de los peces y, en consecuencia, debe ofrecer a éstos comodidad para su desplazamiento y crecimiento. Sólo así podrán mantenerse peces sanos, activos, vistosos y aptos para la reproducción. (F.MORENO, 2004)

El interior del acuario, no es solamente el recipiente de vidrio, sino que incluye los siguientes ítems:

1) Agua
2) Arena o piedras
3) artefactos de calefacción
4) termómetros
5) filtros y aireadores

Todo esto dentro del envase de vidrio. Por fuera se debe mencionar el sistema de iluminación y las bombas de aire.



1 FIGURA 2.1. ACUARIO SIN PLANTAS **(F.MORENO, 2004)**

Los peces se mantienen sanos y fuertes, sin plantas en el acuario, a condición de que se cumplan estrictamente los siguientes requisitos:

1) Fondo sin arena y sumamente limpio,
2) Acuario más bien bajo y ancho, para ofrecer gran superficie de absorción de aire.
3) Buena aireación, para desprender el anhídrido carbónico y distribuir el oxígeno por toda la masa de agua.
4) Cambio semanal de 1/3 del agua, por otra fresca.
5) No elevar excesivamente la temperatura del agua.
6) No exponer el acuario al sol directo ni a mucha luz.

2.2.1 DIMENSIONES QUE DEBE TENER EL ACUARIO

En principio, cuando existe espacio disponible, si se construye un acuario grande los peces tendrán más libertad de movimiento y los filtros, calentadores restarán, proporcionalmente, menos espacio útil. Asimismo, será posible introducir más variedad de especies. Las peleas entre peces serán menores, puesto que la agresividad de muchos peces disminuye en ambientes grandes; a su vez, los peces más chicos dispondrán de mayor espacio para ocultarse. Al mismo tiempo, la temperatura del agua será más estable, puesto que el enfriamiento será más lento que en un acuario pequeño, donde la relación entre la masa de agua y las superficies que están en contacto con el aire es mucho menor. (F.MORENO, 2004)

En general, un acuario comunitario (o sea una pecera para recreación, con peces de todo tipo, destinados a la exposición, no a la cría), requiere un mínimo de 50 litros de capacidad, que se puede obtener, por ejemplo, con las siguientes dimensiones: 60 x 35 x 25 cm. (longitud, altura, fondo, respectivamente), lo que da:

60 x 35 x 25 =52500 cm^3

52500 cm^3 = 52,5 dm^3

Como 1 dm^3 = 1 Litro, la capacidad calculada corresponderá a igual cantidad de litros de agua. Si se descuenta el espacio que debe dejarse libre arriba, para no llegar al tope, el volumen de agua será 3/4 del calculado anteriormente.

En la Tabla 2.1. se muestra tres medidas de acuarios como ejemplo.

PECERA	DIMENSIÓN (cm)	CAPACIDAD (litros)
Pequeña	60x30x24	61.2
Mediana	80x40x55	176.0
Grande	125x48x57	342.0

TABLA 2.1. EJEMPLOS DE MEDIDAS DE ACUARIOS (PROPIO)

2.2.2. FERMENTACIÓN Y PUTREFACCIÓN

Algo que es necesario tener en cuenta, en relación con las dimensiones del acuario, es que las materias fecales de los peces son transformadas en fertilizantes orgánicos por las bacterias, que siempre existen. Si el acuario es grande, la concentración de materias fecales y orina será pequeña, pero si el volumen es reducido, entonces la concentración de los residuos será muy grande, lo que producirá el aumento de la cantidad de bacterias formadas. Como las bacterias respiran, se reducirá el oxígeno disponible para los peces. Además, el agua se tornará lechosa y despedirá mal olor, debido a que la reducción del oxígeno favorece la proliferación de las bacterias anaerobias que producen la fermentación y putrefacción. (F.MORENO, 2004)

2.2.3. EL AGUA

El agua representa para el pez lo que el aire para el resto de los animales; o sea, el medio donde nace, vive, se reproduce y muere. Por lo tanto el agua debe ofrecer determinadas características para que el ciclo vital se realice en óptimas condiciones. Pero estas características son distintas según las regiones del globo terrestre, dependiendo de la naturaleza del terreno sobre el cual corre el líquido, dado que éste disuelve substancias orgánicas y minerales que incorpora. Adicionalmente, se tiene que agregar algo muy importante. Los peces requieren luz y calor adecuados para su crecimiento. Esto lleva, a establecer que el agua que se vierte dentro del recipiente deberá ser elevada a la temperatura considerada normal para el acuario armado. Esto dependerá de la especie de pez que habite el acuario. (F.MORENO, 2004)

Acuario	Temperatura (°C)
Agua Fría	16 – 20
Agua Templada	21 – 24
Agua Caliente	25 – 28

TABLA 2.2 RANGO DE TEMPERATURAS DEL AGUA EN EL ACUARIO

Por lo tanto, el agua deberá ser calentada previamente antes de ser vertida en el acuario, con el fin de no someter a los peces a sufrimiento alguno durante el período de esta etapa.

Calentar el agua es una buena precaución puesto que el desprendimiento del cloro se acelera. La agitación también ayuda al proceso de eliminación del cloro, así como el filtrado a través de carbón activado. (F.MORENO, 2004)

2.2.4. PH DEL AGUA

El agua contiene iones libres negativos de hidroxilo (OH) e iones libres positivos de hidrógeno (H). La acidez de una solución depende de la concentración de los iones de hidrógeno (expresado por el símbolo pH). El agua “neutra” consta de igual cantidad de iones hidroxilos y de iones de hidrógeno, habiéndose determinado, mediante mediciones, que existe un ión de cada tipo por cada 10.000.000 de moléculas (107). Se dice, entonces, que el pH neutro es 7.

Si el agua tuviese la proporción de un ión de H por cada millón de moléculas (106), el pH sería 6, y así sucesivamente. Al aumentar la concentración de un tipo de ion disminuye la del otro.

Los valores de pH varían entre 0 y 14. Un pH de 7 corresponde como se ha dicho a un líquido neutro, por debajo de ese valor será ácido (puesto que la concentración de iones de hidrógeno es mayor) y por arriba, alcalino.

El agua corriente es ligeramente alcalina (pH = 7,3 a 7,9), pero la de los ríos y arroyuelos trópicos, donde viven peces, pueden encontrarse pH de 4,5 a 7. El agua de lluvia tiene un pH = 5.8, por lo que es adecuada para acuarios.

Con una acidez de 6,8 se favorece la secreción de la piel de los peces, protegiéndolos contra la infección. El secreto de conservar muchos alevines radica, generalmente, en el valor correcto del pH en el agua.

El pH del agua de un acuario aumenta durante el día, por la acumulación del ácido carbónico. Sin embargo, en un acuario con exceso de peces, mal aireado, el pH puede bajar a 5,5 o menos en pocas horas.

En general, los peces requieren que el agua sea ligeramente ácida, pero algunos viven mejor si el agua es alcalina. Por este motivo es necesario conocer el método apropiado para establecer la condición imperante en el acuario. Algunas especies toleran amplios cambios de pH, pero otras no. (F.MORENO, 2004)

2.2.5. LA AIREACIÓN

La aireación tiene un único objeto, agitar el agua para que entre oxígeno y se desprenda el anhídrido carbónico. En forma secundaria, las ondulaciones producidas en la superficie dan lugar a un aumento de la misma, permitiendo un mayor intercambio gaseoso, con salida de anhídrido carbónico y absorción de oxígeno.

La agitación del agua evita que el anhídrido carbónico, que es más liviano que el oxígeno, se deposite en las capas superiores, impidiendo la absorción del segundo.

Las burbujas de aire, desprendidas por el aparato aireador, aportan mayor cantidad de oxígeno como hasta hace poco se suponía, partiendo de la base de que aquellas aumentan la superficie del agua en contacto con el aire. Lo que sucede es que el movimiento de la superficie del agua provoca el

desprendimiento del anhídrido carbónico. La agitación del líquido produce el desprendimiento rápido del aire inyectado.

El síntoma más evidente de que la aireación está resultando necesaria, se tiene cuando los peces permanecen con sus hocicos aplicados contra la superficie del agua, boqueando desesperadamente en busca del oxígeno que les falta y que el anhídrido carbónico acumulado impide llegar fácilmente. Con la aireación el anhídrido carbónico se desprende más fácilmente y el oxígeno es distribuido por toda la masa de agua.

La falta de oxígeno, o su mala distribución, permite la proliferación de las bacterias anaerobias, que para respirar descomponen la materia orgánica, liberando oxígeno y produciendo fermentaciones y putrefacciones. (F.MORENO, 2004)

La aireación tiene otras funciones:

a) Mezclar las capas de agua a distintas temperaturas de calentamiento.

b) Destrucción de la película de aceite en la superficie. La película de aceite tiene su origen en la capa de polvo acumulada en la superficie. Tal capa permite el desarrollo de las bacterias anaerobias, debido a que dificultan el intercambio gaseoso del agua con la atmósfera. La aireación rompe esta capa.

c) Evitar la sobresaturación de oxígeno, manteniendo el equilibrio normal de la presión gaseosa. Esto se manifiesta cuando un acuario tiene muchas plantas o una superabundancia de algas.

2.2.6. AIREADORES

Los aparatos aireadores no actúan inyectando oxígeno en el agua, sino agitando ésta y desprendiendo anhídrido carbónico.

2.2.7. FILTRADO DEL AGUA

El filtrado continuo del agua tiene la ventaja adicional de producir corrientes en el líquido, con lo que se distribuye mejor el oxigeno en toda su masa. Así se evita la proliferación de bacterias anaerobias y la putrefacción consiguiente. (F.MORENO, 2004)

2.2.7.1. TIPOS DE FILTROS

Los filtros pueden clasificarse en dos grandes categorías:

a) filtros exteriores
b) filtros interiores.

2.2.7.1.1 FILTROS EXTERIORES

Constan de un envase de plástico, dentro del cual se han introducido capas superpuestas de carbón activado; a veces lana de vidrio o fibras sintéticas. En la parte inferior contienen una cámara de agua, separada de las capas recién mencionadas por una placa perforada.

Ventaja y desventajas del filtro exterior:

La ventaja del filtro exterior es su elevado rendimiento en litros por hora, pero en cambio presenta varios inconvenientes:

- Afecta el aspecto de la instalación, dado que ocupa un lugar en el exterior, junto al acuario.
- Exige tener la misma altura que el acuario, salvo en los modelos de alta presión, cerrados herméticamente. (F.MORENO, 2004)

2. FIGURA 2.2. FILTRO EXTERIOR. **(F.MORENO, 2004)**

2.2.7.1.2. FILTROS INTERIORES

Son los más prácticos para el acuarista aficionado, pudiendo considerarse dos grandes sistemas:

a) No cubierto por arena o piedras.

b) Cubiertos por arena o piedras (mecánicos).

NO CUBIERTO POR ARENA

Posee en su interior una cámara de agua, en un extremo, rellenándose el resto con capas de carbón activado y perlón. El aire inyectado por el aireador produce la salida de burbujas por otro tubo, que emerge verticalmente. La entrada de agua al filtro se produce por el extremo opuesto, de modo que debe pasar por todas las capas sucesivas de carbón y fibra de perlón.

El perlón es un algodón sintético formado con tres tipos diferentes de microfibras entrelazadas: nylon filamento textil, usado en la fabricación de telas; nylon fibra corta, empleado en mezcla con fibras naturales, artificiales y sintéticas, y nylon filamento industrial, empleado por las industrias pesquera, llantera y de cepillos, constituyendo un buen aislante térmico y acústico.

El carbón activado debe ser de grano mediano, no muy fino, a fin de no obturar los conductos del filtro. Esto hace que sólo se detengan las moléculas grandes de las substancias filtradas y pasen las más chicas. (F.MORENO, 2004)

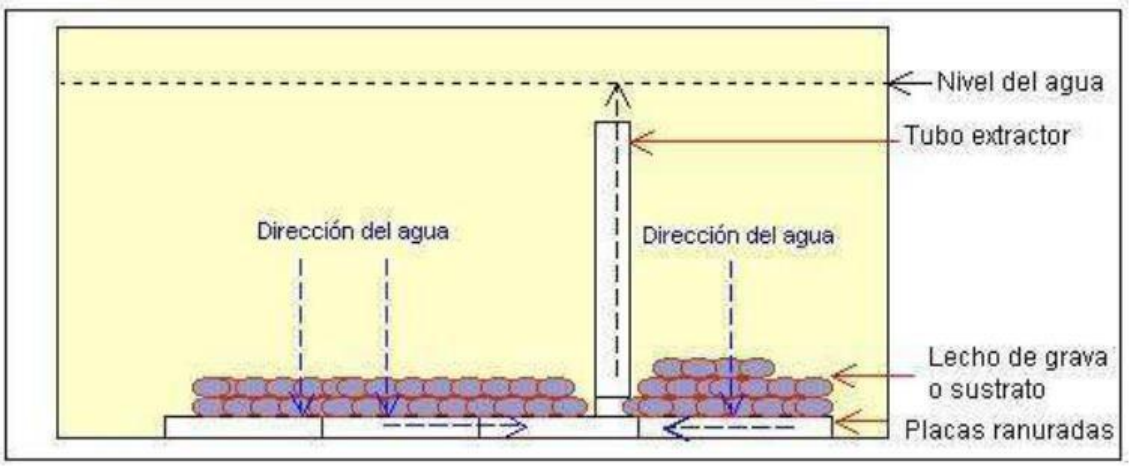
3. FIGURA 2.3 FILTRO INTERIOR NO CUBIERTO POR ARENA **(F.MORENO, 2004)**

CUBIERTOS POR ARENA O PIEDRAS (MECÁNICOS)

La filtración mecánica consiste en la extracción de las partículas sólidas del agua de nuestro acuario. La filtración mecánica no remueve directamente el amoníaco. Los filtros mecánicos comúnmente no remueven las bacterias ni las algas del agua. Tampoco los filtros mecánicos pueden remover sólidos atrapados por la arena, plantas o decorado.

El sistema convencional, por medio del cual el agua circula pasando primero por la arena o las piedras y luego sale a través de los picos o tubos extractores, se ilustra de la siguiente manera:

El aire producido por el aireador impulsa, por succión dentro del tubo, una corriente de agua ascendente. El tubo está firmemente sujeto a la placa y por lo tanto el agua que asciende es tomada desde debajo de la misma. El agua es reemplazada por la que está dentro del acuario. Se produce así un circuito cerrado, en el cual la misma agua pasa varias veces por día a través de la arena o piedras, por debajo de las placas y retorna por el tubo dentro del acuario (líneas azules en la Figura 2.4). (F.MORENO, 2004)



4. FIGURA 2.4. FILTRO INTERIOR CUBIETO POR ARENA O MECANICO **(F.MORENO, 2004)**

A medida que el agua circula entre la arena o piedras, ésta retiene las impurezas, que no son otra cosa que los des hechos orgánicos de los peces, restos de comida, hojas de plantas muertas y otros residuos. En la medida que el filtro se vaya ensuciando éste irá atrapando partículas cada vez más pequeñas, llegando hasta el punto final en que no pasará más agua.

Desventajas de los filtros interiores

Los inconvenientes de los filtros interiores, son los siguientes:

- Es necesario cambiar la fibra de perlón periódicamente, para eliminar la gran cantidad de partículas retenidas (cada mes, por lo menos), evitando así la putrefacción y la consiguiente contaminación del agua.
- Se detienen las bacterias, que son elementos útiles y necesarios para la transformación de los residuos orgánicos en fertilizantes. Se debe colocar un filtro de este tipo cada 50 litros de agua. (F.MORENO, 2004)

2.2.8. TEMPERATURA

Todo acuario con peces tropicales debe calentarse para elevar la temperatura del agua hasta el valor que requieren los peces en ella contenidos, de acuerdo con su naturaleza. La temperatura regula el proceso metabólico del pez, pudiendo establecerse que cuanto mayor sea la temperatura del cuerpo, más elevado será el régimen metabólico y más hambriento estará el pez. Por lo tanto, en un acuario de agua caliente, el pez consumirá más energía.

Por lo expuesto debe cuidarse para no superar la temperatura normal requerida por el pez, según su especie, ya que ello representara una seria perdida de energía y su rápido debilitamiento orgánico. Se considera que a 27 ºC la mayor parte de los peces alcanzan su máximo consumo de oxígeno y apetito.

Los peces están adaptados a las variaciones de temperatura en su medio natural, pero dentro de ciertos rangos. Con esto se quiere decir que no deben permitirse cambios bruscos, pues en este caso podrían producirse resfríos, y afecciones de la vejiga natatoria, con trastornos del equilibrio.

Una variación de 3 ºC hacia abajó puede producir un serio “shock”, enfriamiento y enfermedades. Así, un cambio lento no debe ser mayor de 3 ºC por hora pero no pasando nunca de 3 ºC hacia abajo y 4 ºC hacia arriba. Un pez joven es menos sensible que los adultos a los cambios bruscos de temperatura. Las fluctuaciones del día a la noche no deben ser superiores a 3 ó 4 ºC. (F.MORENO, 2004)

2.2.8.1. CALENTADORES

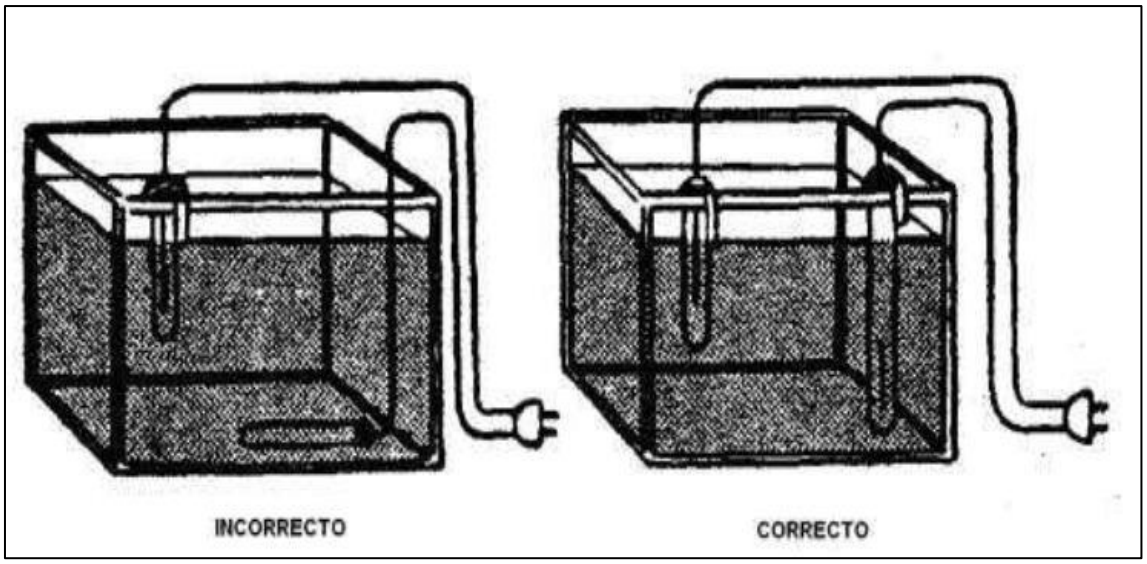
Para la elección del calentador se calcula a razón de 1 watt por litro de agua. El sistema de calentamiento está provisto generalmente de un termostato que regulado, permitirá mantener constante la temperatura del líquido en el acuario a los valores deseados. (F.MORENO, 2004)

***5.FIGURA 2.5. CALENTADOR DE ACUARIO* (F.MORENO, 2004)**

2.2.8.2. COLOCACIÓN DEL CALENTADOR

El calentador debe actuar sobre todo el volumen de líquido; por lo tanto, debe abandonarse la difundida costumbre de disponer el artefacto en la parte superior. Por el contrario, cuanto más cerca del fondo esté el calentador, como se muestra en la Figura 1.6, mejor se calentará la masa total de agua, debido al líquido caliente, siendo menos denso que el frío, tendrá a ascender, produciendo al mismo tiempo corriente en el interior del acuario.

Pero colocar el calentador en la parte inferior, no significa apoyarlo sobre el fondo, puesto a que con el tiempo se cubrirá con un sedimento calcáreo, el que actuará de material aislante térmico. Una buena idea consiste en disponer el calentador cerca de la salida de un aireador; en esa forma el calor se propagará rápidamente al resto del acuario, conducido por la corriente de aire impulsado. Para que la superficie superior del agua no se enfríe rápidamente es necesario colocar una tapa de vidrio. (F.MORENO, 2004)



6. FIGURA 2.6. COLOCACION CORRECTA DE UN CALENTADOR EN UN ACUARIO
(F.MORENO, 2004)

2.2.9. ILUMINACIÓN

El acuario constituye un motivo de decoración, por lo mismo debe estar iluminado de modo tal que permita observar los más mínimos detalles de su interior, el desarrollo de los peces, su estado de salud, su colorido y todo lo que constituye la actividad de ese pequeño mundo encerrado entre las paredes de vidrio.

Pero no solo ese es el fin primordial de la iluminación sino que a falta de luz solar (o con luz solar muy deficiente) se debe suplir la misma, mediante luz artificial no ultravioleta.

La falta de luz favorece la reproducción de las bacterias anaerobias. El resultado de una falta de iluminación será, por lo tanto, peces débiles y descoloridos, agua amarillenta y mal oliente.

La calidad de la luz es importante, ya que contribuye, entre otros factores, al crecimiento sano de los peces. Se recomienda luz de tubo fluorescente de tipo luz de día o equivalente.

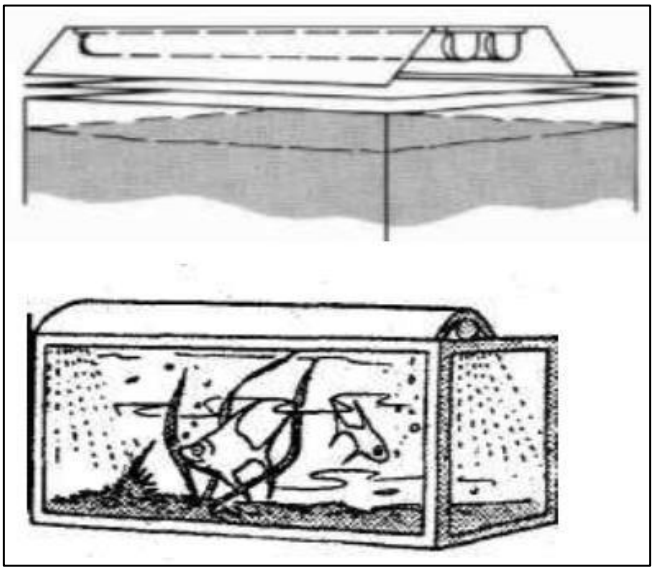
Por estas razones se debe iluminar el acuario por medio de lámparas, controladas por un temporizador, para iluminar durante unas 6 a 9 horas. (F.MORENO, 2004)

2.2.9.1. POSICIÓN DE LAS LÁMPARAS

La lámpara debe colocarse encima del acuario, pero ahí es posible elegir dos posiciones, según lo que se desea destacar:

a) Lámpara bien al frente: es más decorativa, pero deja el fondo en penumbra y otorga sentido de profundidad.

b) Lámpara hacia la parte posterior: excelente iluminación en el interior de la pecera, pero se pierde en vistosidad y el acuario parece reducido en sus dimensiones antero-posterior. Los peces translúcidos, por el contrario, lucen más, puesto que resultan más transparente, visualizándose mejor sus órganos internos. (F.MORENO, 2004)



7.FIGURA 2.7. POSICION E ILUMINACION DE LAMPARAS EN ACUARIOS **(F.MORENO, 2004)**

Se debe tener cuidado de no colocar las lámparas demasiado cerca de la tapa de vidrio, pues el calor la romperá, además de recalentar el agua del acuario en forma peligrosa e incontrolada. Tampoco muy lejos, porqué la iluminación decrece en proporción al cuadrado de la distancia.

2.2.10. ESPECIES DE PECES

Los peces son animales vertebrados acuáticos, generalmente recubiertos en su mayoría por escamas y dotados de aletas, que permiten su desplazamiento en el medio acuático. Los peces son abundantes tanto en agua salada como en agua dulce, pudiéndose encontrar especies desde los arroyos de montaña, así como en lo más profundo del océano. (F.MORENO, 2004)

Algunas de las especies de peces más comunes que se encuentran en la Ciudad de Piura son:

VIEJA (ANDINOCARA)

8. FIGURA 2.8 VIEJA (ANDINOCARA).

Características:

- Temperatura: 20ºC - 25ºC
- pH del agua: 6.5 – 7.5
- Comportamiento: comunitario
- Origen: Panamá, Colombia, Ecuador y Perú

CHAME (DORMITATOR LATIFRONS)

9. FIGURA 2.9. CHAME (DORMITATOR LATIFRONS).

Características:

- Temperatura: 21ºC - 28ºC
- pH del agua: 6.5 - 9.0
- Comportamiento: comunitario
- Origen: Norte América, Ecuador, Perú

ESCALAR (PTEROPHYLLUM SCALARE)

10. FIGURA 2.10 ESCALAR (PTEROPHYLLUM SCALARE).

Características:

- Temperatura: 24ºC -28ºC
- pH del agua: 6.0 - 7.5
- Comportamiento: comunitario
- Origen: América del Sur

GUPPY (POECILIA RETICULATA)

11. FIGURA 2.11 GUPPY (POECILIA RETICULATA).

Características:

- Temperatura: 24ºC -28ºC
- pH del agua: 6.0 - 8.0
- Comportamiento: comunitario
- Origen: América del Sur

CÍCLIDOS (BUJURQUINA SYSPILUS)

12.FIGURA 2.12 CÍCLIDO (BUJURQUINA SYSPILUS).

Características:

- Temperatura: 20ºC -26ºC
- pH del agua: 7.5 – 8.5
- Comportamiento: comunitario
- Origen: África y América del Sur

OSCAR (ASTRONOTUS OCELLATUS)

13. FIGURA 2.13. OSCAR (ASTRONOTUS OCELLATUS).

Características:

- Temperatura: 24ºC -28ºC
- pH del agua: 6.0 – 8.0
- Comportamiento: agresivo
- Origen: Brasil, Paraguay

CARACHAMA (LORICARIIDAE)

 .

14. FIGURA 2.14. CARACHAMA (LORICARIIDAE).

Características:

- Temperatura: 18ºC -26ºC
- pH del agua: 6.0 – 8.0
- Comportamiento: comunitario
- Origen: América del Sur

GURAMI AZUL (TRICHOGASTER TRICHOPTERUS)

15. FIGURA 2.15. GURAMI (TRICHOGASTER TRICHOPTERUS).

Características:

- Temperatura: 22ºC -28ºC
- pH del agua: 6.0 - 8.0
- Comportamiento: comunitario
- Origen: Asia, Malaysia, Vietnam, Tailandia

GOLDFISH (CARASSIUS AURATUS)

16.FIGURA 2.16. GOLDFISH (CARASSIUS AURATUS).

Características:

- Temperatura: 15ºC -24ºC
- pH del agua: 6.0 – 8.5
- Comportamiento: comunitario
- Origen: Asia

ESPADITA (XIPHOPHORUS HELLERI)

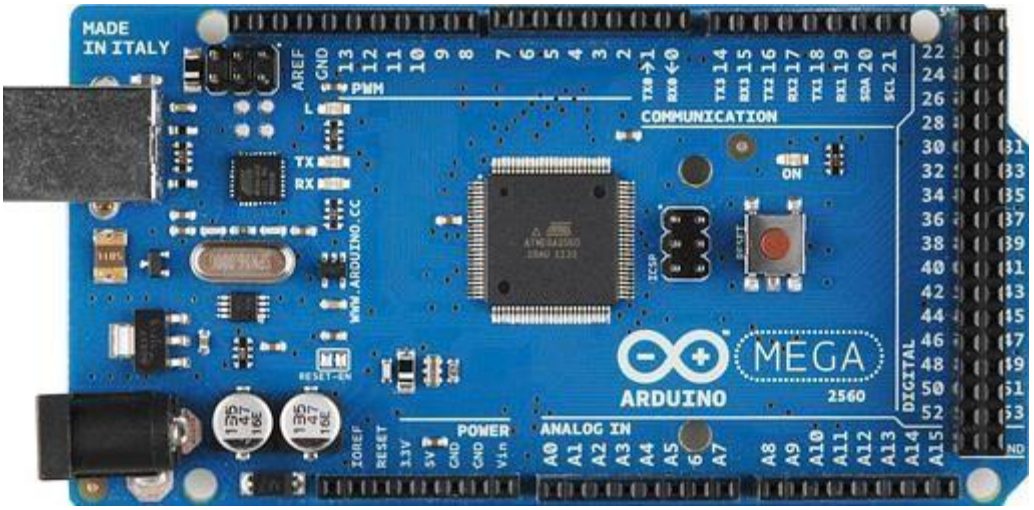
17.FIGURA 2.17. ESPADITA (XIPHOPHORUS HELLERI).

Características:

- Temperatura: 20ºC -24ºC
- pH del agua: 7.0 – 8.0
- Comportamiento: comunitario
- Origen: Centroamérica

2.3. ARDUINO MEGA 2560

La placa ARDUINO MEGA 2560 es la que se muestra en la siguiente Figura 2.3.



18.FIGURA 2.18. ARDUINO MEGA 2560 (www.robotshop.com)

2.3.1. VISION GENERAL

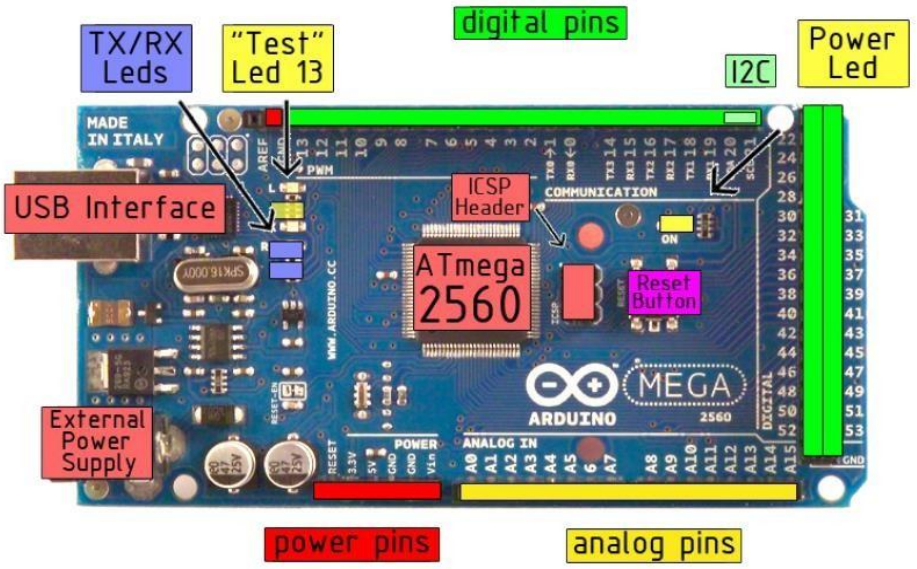
El ARDUINO MEGA 2560 es una placa basada en el microprocesador Atmega2560. Tiene 54 entradas/salidas digitales (de las cuales 15 pueden utilizarse para salidas PWM), 16 entradas analógicas, 4 UARTs (puertos serie por hardware), un oscilador de 16MHz, una conexión USB, entrada de corriente, conector ICSP y botón de RESET. Contiene todo lo necesario para hacer funcionar el microcontrolador, simplemente conectarlo a un ordenador con un cable USB, o alimentarlo con un adaptador de corriente AC a DC para empezar. Mega es compatible con la mayoría de los SHIELD diseñados para ARDUINO DUEMILANOVE O DIECIMILA.

MEGA 2560 es una actualización de la placa ARDUINO MEGA, al que sustituye. MEGA 2560 difiere de todas las placas anteriores ya que no utiliza el chip controlador de USB a serial FTDI. En su lugar, ofrece el ATMEGA16U2 programado como convertidor USB a serie. Esto implica que se trabaje con /dev/ACM. (ATMEL, s.f.)

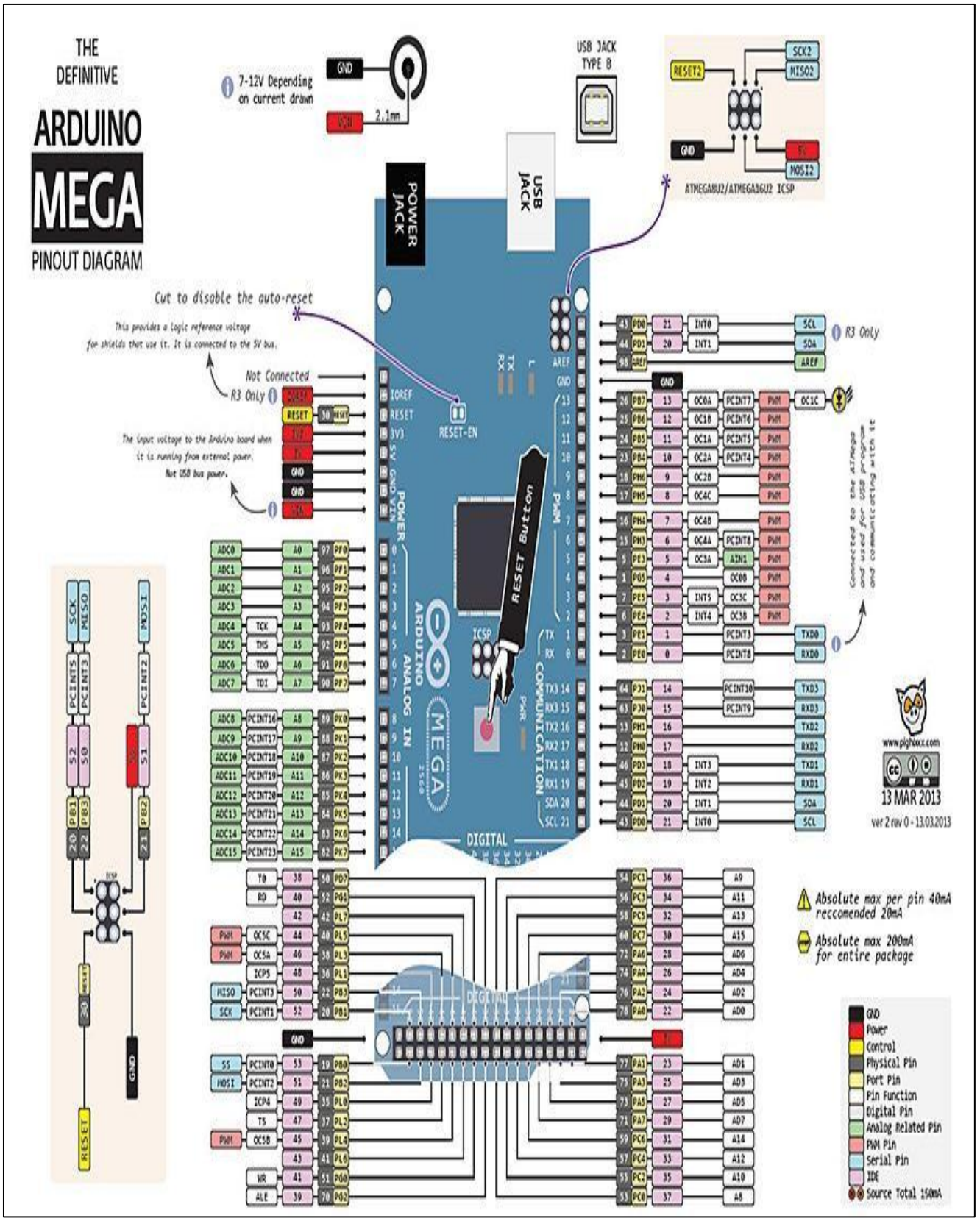
2.3.2. MAPEADO DE PIN

2.3.3. RESUMEN

- Microcontrolador ATmega2560
- Voltaje de funcionamiento 5V
- Voltaje de entrada (recomendado) 7-12V
- Voltaje de entrada (limite) 6-20V
- Pines E/S digitales 54 (15 proporcionan salida PWM)
- Pines de entrada analógica 16
- Intensidad por pin 40 mA
- Intensidad en pin 3.3V 50 mA
- Memoria Flash 256 KB de las cuales 8 KB las usa el gestor de arranque (bootloader)
- SRAM 8 KB
- EEPROM 4 KB
- Velocidad de reloj 16 MHz

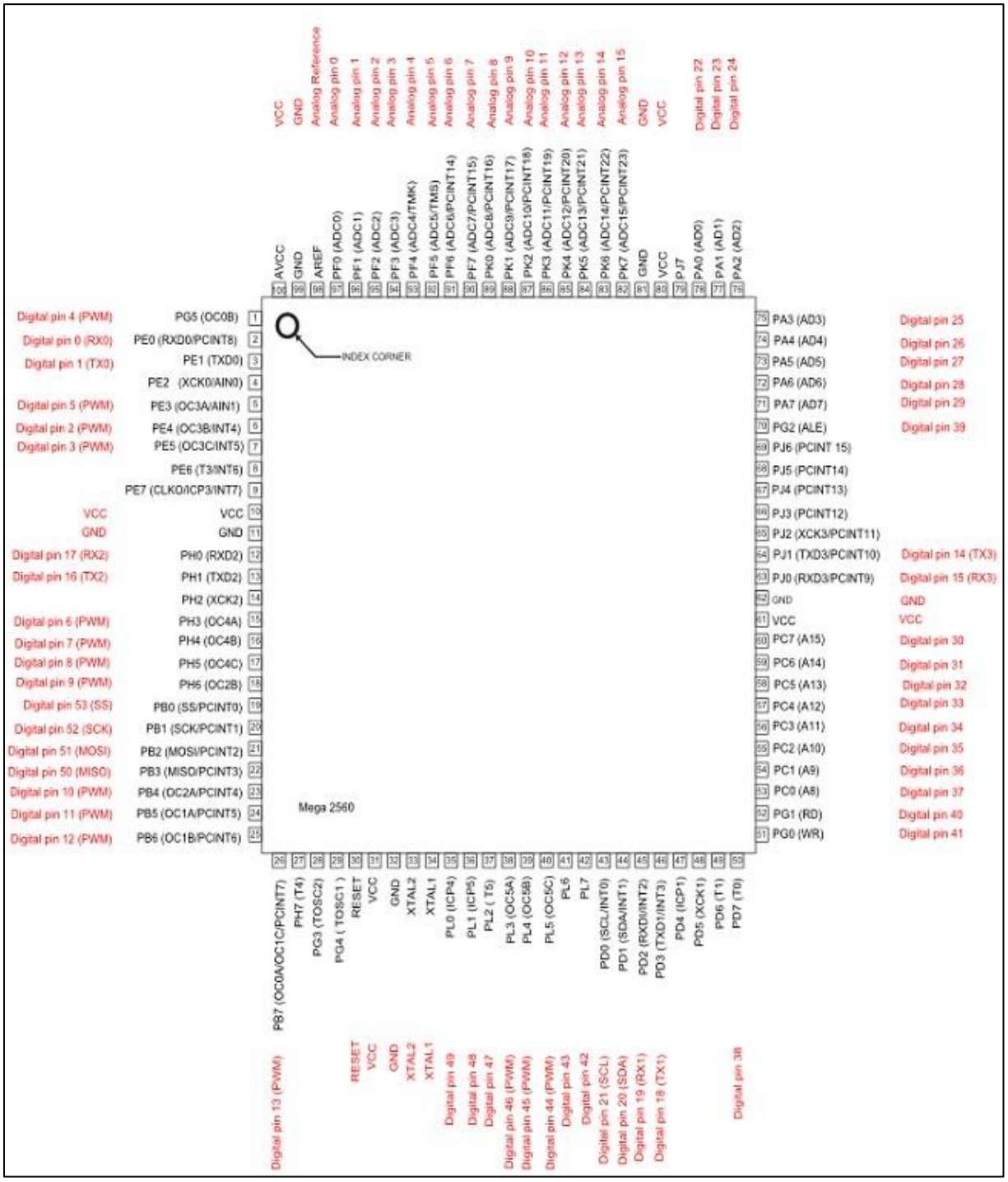


19.FIGURA 2.19. DISTRIBUCION DE PINES DEL ARDUINO MEGA 2560 (www.robotshop.com)



(F.MORENO, 2004)

20.FIGURA 2.20. DIAGRAMA DE PINES DEL ARDUINO MEGA 2560. (www.robotshop.com)



21.FIGURA 2.21. DIAGRAMA DE PINES DEL MICROCONTROLADOR MEGA 2560. (www.robotshop.com)

2.3.4 ALIMENTACIÓN

El ARDUINO MEGA puede ser alimentado por la conexión USB o por una fuente de alimentación externa. El origen de la alimentación es seleccionado automáticamente. (ATMEL, s.f.)

Las fuentes de alimentación externas (no-USB) pueden ser un transformador o una batería. El transformador se puede conectar usando un conector macho de 2.1mm con centro positivo en el conector hembra de la placa. Los cables de la batería pueden conectarse a los pines GND y Vin en los conectores de alimentación (POWER).

La placa puede trabajar con una alimentación externa de entre 6 a 20 voltios. Si el voltaje suministrado es inferior a 7V el pin de 5V puede proporcionar menos de 5 Voltios y la placa puede volverse inestable, si se usan mas de 12V los reguladores de voltaje se pueden sobrecalentar y dañar la placa. El rango recomendado es de 7 a 12 voltios. (ATMEL, s.f.)

Los pines de alimentación son los siguientes:

- **VIN.** La entrada de voltaje a la placa ARDUINO cuando se está usando una fuente externa de alimentación (en contraposición a los 5 voltios de la conexión USB). Se puede proporcionar voltaje a través de este pin, o, si se está alimentado a través de la conexión de 2.1mm, acceder a ella a través de este pin.
- **5V.** La fuente de voltaje estabilizado usado para alimentar el microcontrolador y otros componentes de la placa. Esta puede provenir de VIN a través de un regulador integrado en la placa, o proporcionada directamente por el USB u otra fuente estabilizada de 5V.
- **3V3.** Una fuente de voltaje a 3.3 voltios generada en el chip FTDI integrado en la placa. La corriente máxima soportada 50mA.
- **GND.** Pines de toma de tierra.
- **IOREF.** Este pin proporciona la referencia de tensión con la que opera el microcontrolador. Un shield configurado puede leer el voltaje pin IOREF y seleccionar la fuente de alimentación adecuada o habilitar traductores tensión en las salidas para trabajar con los 5V o 3.3V. (ATMEL, s.f.)

2.3.5 ENTRADAS Y SALIDAS

Cada uno de los 54 pines digitales en el Mega pueden utilizarse como entradas o salidas usando las funciones pinMode(), digitalWrite(), y digitalRead() . Las E/S operan a 5 voltios. Cada pin puede proporcionar o recibir una intensidad máxima de 40mA y tiene una resistencia interna (desconectada por defecto) de 20-

50kOhms. Además, algunos pines tienen funciones especializadas: (ATMEL, s.f.)

- **SERIE:** 0 (RX) y 1 (TX), Serie 1: 19 (RX) y 18 (TX); Serie 2: 17 (RX) y 16 (TX); Serie 3: 15 (RX) y 14 (TX). Usado para recibir (RX) transmitir (TX) datos a través de puerto serie TTL. Los pines Serie: 0 (RX) y 1 (TX) están conectados a los pines correspondientes del chip ATmega16U2.
- **INTERRUPCIONES EXTERNAS:** 2 (interrupción 0), 3 (interrupción 1), 18 (interrupción 5), 19 (interrupción 4), 20 (interrupción 3), y 21 (interrupción 2). Estos pines se pueden configurar para lanzar una interrupción en un valor LOW (0V), en flancos de subida o bajada (cambio de LOW a HIGH (5V) o viceversa), o en cambios de valor.
- **PWM:** de 2 a 13 y 44 a 46. Proporciona una salida PWM (Pulse Wave Modulation, modulación de onda por pulsos) de 8 bits de resolución (valores de 0 a 255) a través de la función analogWrite().
- **SPI:** 50 (MISO), 51 (MOSI), 52 (SCK), 53 (SS). Estos pines soportan comunicación SPI utilizando la biblioteca de SPI . Los pines SPI también se desglosan en la cabecera ICSP, que es físicamente compatible con el Uno, Duemilanove y Diecimila.
- **LED:** 13. Hay un LED integrado en la placa conectado al pin digital 13, cuando este pin tiene un valor HIGH(5V) el LED se enciende y cuando este tiene un valor LOW(0V) este se apaga.
- **TWI:** 20 (SDA) y 21 (SCL) Apoyar la comunicación TWI utilizando la librería Wire.
- El Mega2560 tiene 16 entradas analógicas, cada una de las cuales proporcionan una resolución de 10 bits (1.024 valores diferentes). Por defecto se miden desde el tierra a 5 voltios, aunque es posible cambiar la cota superior de su rango utilizando el pin AREF y función analogReference ().

  Hay unos otros pines en la placa:
- **AREF.** Voltaje de referencia para las entradas analógicas. Usado por analogReference(). (ATMEL, s.f.)

- **RESET.** Suministrar un valor LOW (0V) para reiniciar el microcontrolador. Típicamente usado para añadir un botón de reset a los shields que no dejan acceso a este botón en la placa.

2.3.6. COMUNICACIÓN

El **ARDUINO MEGA2560** tiene una serie de facilidades para la comunicación con un ordenador, otro **ARDUINO,** u otros microcontroladores. El **ATMEGA2560** proporciona cuatro UART hardware para TTL (5V) de comunicación serie. Un **ATMEGA16U2** canaliza a uno de ellos sobre el USB y proporciona un puerto com virtual para software al equipo (máquinas de Windows tendrá un archivo .inf, pero las máquinas OSX y Linux reconocerán la placa automáticamente). El **SOFTWARE DE ARDUINO** incluye un monitor serie que permite enviar datos desde y hacia la placa. Los LEDs RX y TX de la placa parpadearán cuando se están transmitiendo datos a través de ATmega8U2/ATmega16U2 chip y la conexión USB al ordenador (pero no para la comunicación serial en los pines 0 y 1).

La biblioteca SoftwareSerial permite la comunicación en serie en cualquiera de los pines digitales del **MEGA2560.**

El **ATMEGA2560** también soporta TWI y la comunicación SPI. (ATMEL, s.f.)

2.3.7. PROGRAMACIÓN

El **ATMEGA2560** en el **ARDUINO MEGA** viene precargado con un gestor de arranque **(BOOTLOADER)** que permite cargar nuevo código sin necesidad de un programador por hardware externo. Se comunica utilizando el protocolo STK500 original.

También se puede saltar el gestor de arranque y programar directamente el microcontrolador a través del puerto ISCP (In Circuit Serial Programming). (ATMEL, s.f.)

2.3.8. REINICIO AUTOMÁTICO POR SOFTWARE

En vez de necesitar reiniciar presionando físicamente el botón de reset antes de cargar, el Arduino Mega2560 está diseñado de manera que es posible reiniciar por software desde el ordenador al que esté conectado. Una de las líneas de control de flujo (DTR) del ATmega8U2 está conectada a la línea de reinicio del

ATmega2560 a través de un condensador de 100 nanofaradios. Cuando la línea se pone a LOW (0V), la línea de reinicio también se pone a LOW el tiempo suficiente para reiniciar el chip. El software de Arduino utiliza esta característica para permitir cargar los sketches con solo apretar un botón del entorno. Dado que el gestor de arranque tiene un lapso de tiempo para ello, la activación del DTR y la carga del sketch se coordinan perfectamente.

Esta configuración tiene otras implicaciones. Cuando el Mega2560 se conecta a un ordenador con Mac OS X o Linux, este reinicia la placa cada vez que se realiza una conexión desde el software (vía USB). El medio segundo aproximadamente posterior, el gestor de arranque se ejecutará. A pesar de estar programado para ignorar datos mal formateados (ej. cualquier cosa que la carga de un programa nuevo) intercepta los primeros bytes que se envían a la placa justo después de que se abra la conexión. Si un sketch que se está ejecutando en la placa recibe algún tipo de configuración inicial u otro tipo de información al inicio del programa, asegúrese de que el software con el cual se comunica espera un segundo después de abrir la conexión antes de enviar los datos.

El Mega2560 contiene una pista que puede ser cortada para deshabilitar el auto-reset. Las terminaciones a cada lado pueden ser soldadas entre ellas para rehabilitarlo. Están etiquetadas con "RESET-EN". También se puede deshabilitar el auto-reset conectando una resistencia de 110 Ω desde el pin 5V al pin de reset. (ATMEL, s.f.)

2.3.9. CARACTERÍSTICAS FÍSICAS Y COMPATIBILIDAD DE SHIELDS

La longitud y amplitud máxima de la placa Mega2560 es de 4 y 2.1 pulgadas respectivamente, con el conector USB y la conexión de alimentación sobresaliendo de estas dimensiones. Tres agujeros para fijación con tornillos permiten colocar la placa en superficies y cajas. Se debe tener en cuenta que la distancia entre los pines digitales 7 y 8 es 160 mil (0,16").

El Mega está diseñado para ser compatible con la mayoría de shields diseñados para el Uno, Diecimila o Duemilanove. Pines digitales 0 a 13 (y los pines AREF y GND adyacentes), las entradas analógicas de 0 a 5, los conectores de alimentación, y los conectores ICSP están todos ubicados en posiciones equivalentes. Además la UART principal (puerto serie) se encuentra en los mismos pines (0 y 1), al igual que las interrupciones externas 0 y 1 (pines 2 y 3,

respectivamente). SPI está disponible a través de la cabecera ICSP tanto en el Mega2560 y Duemilanove / Diecimila. (ATMEL, s.f.)

2.3.10. REVISIONES

El Mega 2560 no utiliza el chip controlador FTDI USB-a-serie utilizados en los diseños anteriores. En lugar de ello, se cuenta con el ATmega16U2 (placas Arduino ATmega8U2 revisión 1 y 2) programado como un convertidor de USB a serie.

La revisión 2 de la placa Mega 2560 tiene una resistencia que lleva la línea 8U2 HWB a tierra, por lo que es más fácil de poner en modo DFU.

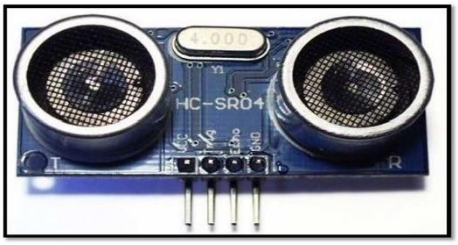
La revisión 3 de la placa ARDUINO y la actual GENUINO MEGA 2560 tienen las siguientes características mejoradas:

- Pines1.0 : Pines SDA y SCL - cerca del pin AREF - y otros dos nuevos pines colocados cerca del pin de RESET,el IOREF que permite a los escudos adaptarse a la tensión suministrada desde la placa. En el futuro, los escudos serán compatibles tanto con la placa que utilice el AVR, que operan con 5 V como con la placa que utiliza ATSAM3X8E, que opera con 3.3 V. El segundo es un pin no está conectado, que está reservado para usos futuros.
- Circuito de RESET más vigoroso.
- ATmega 16U2 sustituye al 8U2. (ATMEL, s.f.)

2.4. SENSOR DE NIVEL HC-SR04

En la figura 2.22 se muestra un sensor ultrasónico. Los ultrasonidos son una radiación mecánica de frecuencia superior a los audibles (20Khz). Toda radiación al incidir sobre un objeto, en parte se refleja, en parte se transmite y en parte es absorbida. Si además hay un movimiento relativo entre la fuente de radiación y el reflector, se produce un cambio de frecuencia de la radiación (Efecto Doppler).Todas estas propiedades de la interacción de una radiación con un objeto han sido aplicadas en mayor o menor grado a la medida de diversas magnitudes físicas. El poder de penetración de la radiación permite que muchas de estas aplicaciones sean totalmente no invasivas, es decir, que no acceda al interior del recinto donde se producen los cambios que se desean detectar. (RODRIGUEZ, 2017)

En función del tiempo que tarda el sonido en rebotar y volver, se calcula la distancia a la que se encuentra dicho objeto.

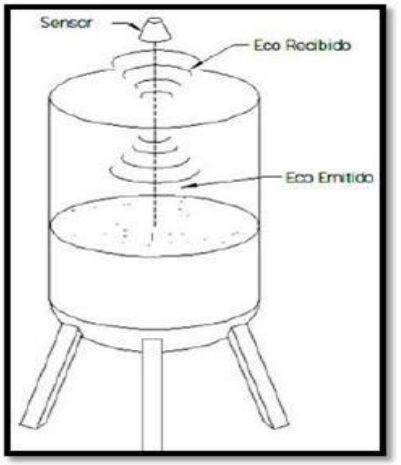


22.FIGURA 2.22 SENSOR DE ULTRASONIDO HC-SR04 **(RODRIGUEZ, 2017)**

2.4.1 FUNCIONAMIENTO DEL SENSOR ULTRASÓNICO

El ultrasonido es sonido exactamente igual al que escucha el ser humano normalmente, pero con una frecuencia mayor a la máxima audible por el oído humano. Ésta comienza desde unos 16 Hz y tiene un límite superior de aproximadamente 20 KHz, mientras que se va a utilizar sonido con una frecuencia de 40 KHz. A este tipo de sonidos es a lo que se denomina Ultrasonidos.

El funcionamiento básico de los sensores ultrasónicos como medidores de distancia se muestra en la figura 2.23, donde se tiene un receptor que emite un pulso de ultrasonido, el cual rebota sobre un determinado objeto y la reflexión de ese pulso es detectada por un receptor de ultrasonidos. (RODRIGUEZ, 2017)



23.FIGURA 2.23 FUNCIONAMIENTO SENSOR DE ULTRASONIDO **(RODRIGUEZ, 2017)**

La mayoría de los sensores de ultrasonido de bajo costo se basan en la emisión de un pulso de ultrasonido cuyo lóbulo, o campo de acción, es de forma cónica.

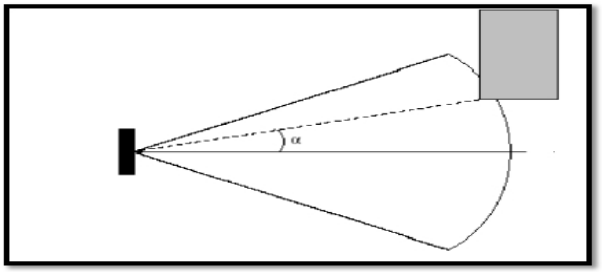
Midiendo el tiempo que transcurre entre la emisión del sonido y la percepción del eco se puede establecer la distancia a la que se encuentra el obstáculo que ha producido la reflexión de la onda sonora, mediante la ecuación:

$$d = \frac{1}{2} V \cdot t$$

Donde V es la velocidad del sonido en el aire y t es el tiempo transcurrido entre la emisión y recepción del pulso.

A pesar de que su funcionamiento parece muy sencillo, existen factores inherentes tanto a los ultrasonidos como al mundo real, que influyen de una forma determinante en las medidas realizadas. Por tanto, es necesario un conocimiento de las diversas fuentes de incertidumbre que afectan a las medidas para poder tratarlas de forma adecuada, minimizando su efecto en el conocimiento del entorno que se desea adquirir. Entre los diversos factores que alteran las lecturas que se realizan con los sensores de ultrasonido cabe destacar:

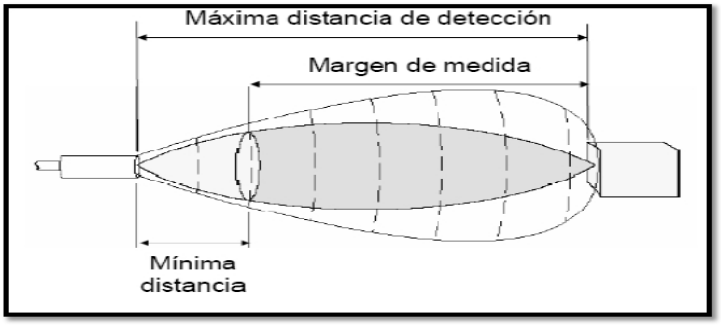
El campo de actuación del pulso que se emite desde un transductor de ultrasonido tiene forma cónica. El eco que se recibe como respuesta a la reflexión del sonido indica la presencia del objeto más cercano que se encuentra dentro del cono acústico y no especifica en ningún momento la localización angular del mismo, como se muestra en la figura 2.24. Aunque la opción de que el objeto detectado esté sobre el eje central del cono acústico, la posibilidad que el eco se haya producido por un objeto presente en la periferia del eje central no es en absoluto despreciable y ha de ser tenida en cuenta y tratada convenientemente (RODRIGUEZ, 2017)



24.Figura 2.24. Incertidumbre angular en la medida de un ultrasonido **(RODRIGUEZ, 2017)**

La cantidad de energía acústica reflejada por el obstáculo depende en gran medida de la estructura de su superficie. Para obtener una reflexión altamente difusa del obstáculo, el tamaño de las irregularidades sobre la superficie reflectora debe ser comparable a la longitud de onda de la onda de ultrasonido incidente.

En los sensores de ultrasónicos de bajo coste se utiliza el mismo transductor como emisor y receptor. Tras la emisión del ultrasonido se espera un determinado tiempo a que las vibraciones en el sensor desaparezcan y esté preparado para recibir el eco producido por el obstáculo. Esto implica que existe una distancia mínima d (proporcional al tiempo de relajación del transductor) a partir de la cual el sensor mide con precisión. Por lo general, todos los objetos que se encuentren por debajo de esta distancia, d, serán interpretados por el sistema como que están a una distancia igual a la distancia mínima, como se muestra en la figura 2.25.



25. FIGURA 2.25 MÁRGENES DE DETECCIÓN DE UN SENSOR ULTRASÓNICO
(RODRIGUEZ, 2017)

Los factores ambientales tienen una gran repercusión sobre las medidas ya que las ondas de ultrasonido se mueven por un medio material que es el aire.

La densidad del aire depende de la temperatura, influyendo este factor sobre la velocidad de propagación de la onda según la expresión:

$$V_S = V_{SO}\sqrt{1 + \frac{T}{273}}$$

Siendo Vso la velocidad de propagación de la onda sonora a 0 ºC, y T la temperatura absoluta (grados Kelvin). (RODRIGUEZ, 2017)

2.4.2 DESCRIPCIÓN DEL FUNCIONAMIENTO EN DETALLE DEL SENSOR HC-SR04

Es un sensor de distancias por ultrasonidos desarrollado por la firma DEVANTECH Ltda. Capaz de detectar objetos y calcular la distancia a la que se encuentra en un rango de 1,7 a 431 cm.

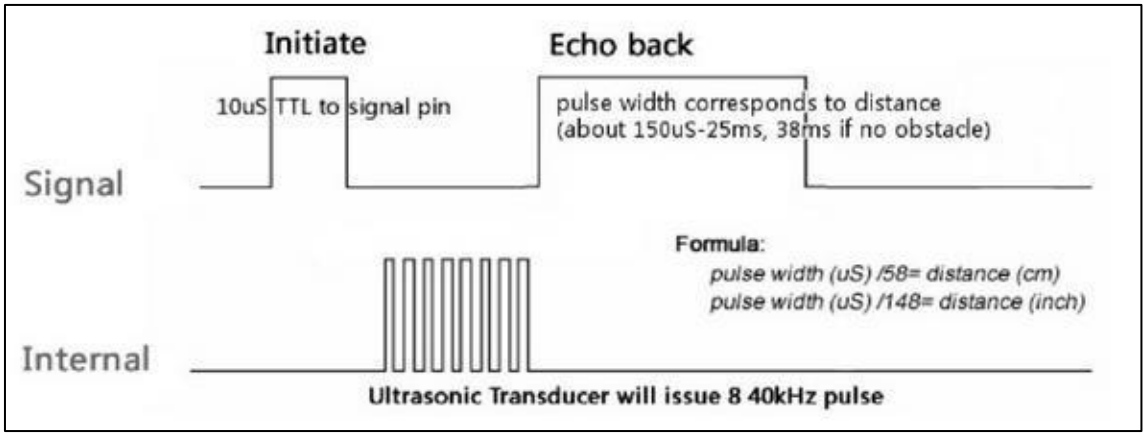
El sensor funciona por ultrasonidos y contiene toda la electrónica encargada de hacer la medición. Su uso es tan sencillo como enviar el pulso de arranque y medir la anchura del pulso de retorno. De muy pequeño tamaño, el sensor se destaca por su bajo consumo, gran precisión y bajo precio.[17]

El sensor funciona emitiendo impulsos de ultrasonidos inaudibles para el oído humano. Los impulsos emitidos viajan a la velocidad del sonido hasta alcanzar un objeto, entonces el sonido es reflejado y captado de nuevo por el receptor de ultrasonidos. Lo que hace el controlador incorporado es emitir una ráfaga de impulsos y a continuación empieza a contar el tiempo que tarda en llegar el eco. Este tiempo se traduce en un pulso de eco de anchura proporcional a la distancia a la que se encuentra el objeto. Registrando la duración del pulso es posible calcular la distancia en pulgadas, centímetros o en cualquier otra unidad de medida. Si no se detecta nada, entonces el HC-SR04 baja el nivel lógico de su línea de eco después de 30mS.[17]

El HC-SR04 proporciona un pulso de eco proporcional a la distancia. Si el ancho del pulso se mide en µS, el resultado se debe dividir entre 58 para saber el equivalente en centímetros, y entre 148 para saber el equivalente en pulgadas. µS/58=cm o µS/148=pulgadas. El HC-SR04 puede activarse cada 50mS, o 20 veces por segundo. Debería esperar 50ms antes de la siguiente activación, incluso si el HC-SR04 detecta un objeto cerca y el pulso del eco es más corto. De esta manera se asegura que el "bip" ultrasónico ha desaparecido completamente y no provocará un falso eco en la siguiente medición de distancia

Desde un punto de vista práctico, lo que hay que hacer es mandar una señal de arranque en el pin 3 del HC-R04 y después leer el ancho del impulso que proporciona en el pin 2. Externamente se aplica, por parte del usuario, un pulso

de disparo o trigger de 10 µS de duración mínima. Se inicia la secuencia. El módulo transmite un tren de pulsos o “burst” de 8 ciclos a 40KHz. En ese momento la señal de salida ECO pasa a nivel “1”. Cuando la cápsula receptora recibe la señal transmitida como consecuencia de haber rebotado en un objeto (eco), esta salida pasa de nuevo a nivel “0”. El usuario debe medir la duración del pulso de esta señal, es decir, el tiempo en que la señal eco se mantiene a “1”. Con objeto de que el módulo se estabilice, se debe dejar un lapsus de tiempo de unos 20ms mínimo entre el momento en que la señal de eco pasa a “0” y un nuevo pulso de disparo que inicie el siguiente ciclo o medida. Esto permite realizar medidas cada 50ms o lo que es igual a 20 medidas por segundo. La duración del pulso eco de salida varía entre 100µs y 25ms, en función de la distancia entre las cápsulas del módulo y el objeto. La velocidad del sonido es de 29,15 µs/cm que, como realiza un recorrido de ida y vuelta, queda establecida en 58,30µs/cm. Así pues el rango mínimo que se puede medir es de 1,7 cm (100µs/58) y el máximo de 431 cm (25ms/58). En la Figura 2.26 se muestra diagrama de tiempos. (RODRIGUEZ, 2017)



26.FIGURA 2.26 DIAGRAMA DE TIEMPOS DEL SENSOR HC-SR04. **(RODRIGUEZ, 2017)**

2.5. SENSOR DE TEMPERATURA DS18B20

El sensor de temperatura es un dispositivo que permite conocer el valor de temperatura presente en un ambiente acuático, a través de la conversión de los cambios de temperatura a señales eléctrica, esta información es procesada por dispositivos electrónicos según la necesidad, como es el caso del ARDUINO MEGA 2560. (EL-KHOURI, 2016)

Es un dispositivo creado por la compañía Maxim Integrated, que tiene la capacidad de comunicarse a través de señal digital hacia el elemento electrónico

que necesita la obtención de la temperatura. Cuenta con tres terminales: Vcc, GND y el pin Data.

El sensor utiliza la comunicación OneWire para envío de datos a través de un solo hilo, a diferencia de otros dispositivos que utilizan protocolos de comunicación de dos vías (Rx/Tx).

El sensor puede ser conectado a cualquier dispositivo microcontrolador tal como ARDUINO directamente, sin embargo la comunicación entre estos es a través del protocolo One Wire (1-wire) diseñado por Dallas semiconductor, en nuestro caso, el sensor de temperatura se va a conectar a la placa de ARDUINO MEGA 2560.

El sensor utilizado es una versión impermeabilizada del sensor de temperatura DS18B20. El sensor resiste hasta 125 ° C y el cable está encamisado en PVC por lo que sugerimos mantenerla por debajo de 100 ° C.

Debido a que son digitales, no se recibe ninguna señal de degradación, incluso a través de largas distancias. (EL-KHOURI, 2016)

27.FIGURA 2.27 SENSOR DE TEMPERATURA DS18B20 **(EL-KHOURI, 2016)**

El DS18B20 Proporciona 9 a 12 bits lecturas de temperatura (configurable) a través de un interfaz 1-Wire, por lo que las necesidades son solo un conductor (y tierra) para ser conectado a un microprocesador central a la tensión de 3.0-5.5V.

2.5.1. ESPECIFICACIONES TECNICAS DEL SENSOR DS18B20

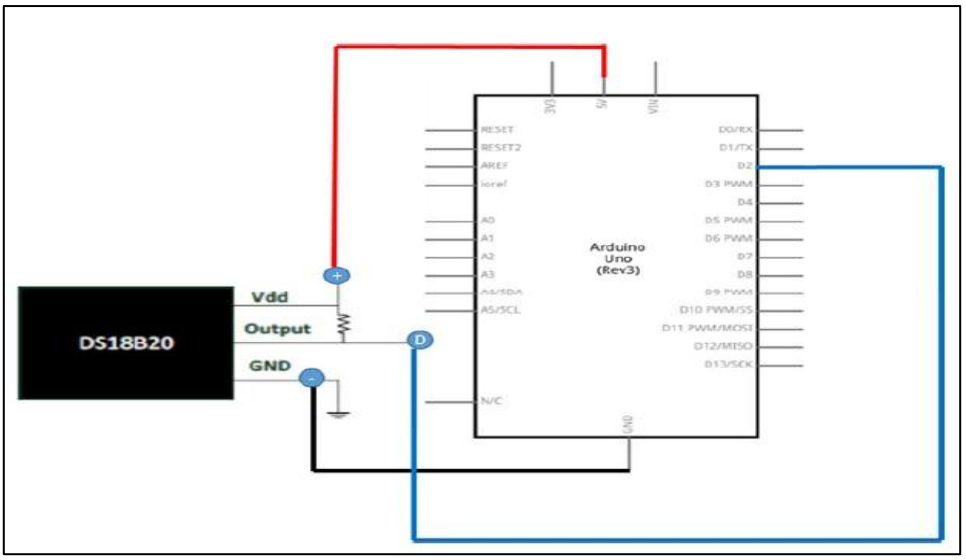
VOLTAJE DE FUNCIONAMIENTO:	3.0 ~ 5.5V.
PRECISIÓN	± 0,5 ° C de -10 ° C a + 85 ° C
UTILIZABLE RANGO DE TEMPERATURA	-55 a 125 ° C (-67 ° F a + 257 ° F)
RESOLUCIÓN SELECCIONABLE	9 a 12 bits
UTILIZA 1-WIRE INTERFAZ	Requiere sólo un pasador para la comunicación digital
SENSORES MÚLTIPLES	Pueden compartir una clavija
SISTEMA DE ALARMA-LÍMITE DE TEMPERATURA	
TIEMPO DE CONSULTA	Menos de 750 ms
CABLES	3 cables de interfaz
	Cable rojo – VCC Cable amarillo – GND El cable verde - DATOS
DIÁMETRO DEL CABLE	4 mm (0,16 ")
INOXIDABLE DE DIÁMETRO DE TUBO DE ACERO	de 6 mm por 35 mm (1,34 ") de largo
LONGITUD	90 cm (35.43 ")

TABLA 2.1 ESPECIFICACIONES TECNICAS DEL SENSOR DS18B20

2.5.2. DIAGRAMA DE BLOQUES

Para lograr el objetivo de monitorear en tiempo real la temperatura, en la figura siguiente se presenta a manera de ejemplo el diagrama a bloques de acoplamiento entre el sensor DS18B20 y la placa ARDUINO UNO R3 a través del pin Digital número D2 con una comunicación 1-wire.

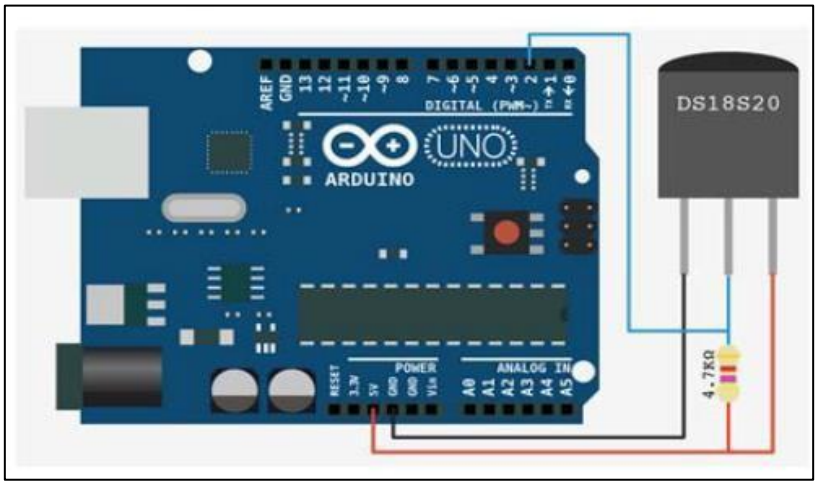
Considerando las especificaciones del fabricante del sensor se coloca una resistencia de 4,7Kohms entre la entrada de energía y la salida de datos del sensor con el fin de mantener un correcto funcionamiento



28.FIGURA 2.28. DIAGRAMA DE BLOQUE DE CONEXIÓN DEL SENSOR DS18B20 CON ARDUINO UNO (PROPIO)

2.5.3. ESQUEMA DE CONEXIONADO DEL DS18B20 CON ARDUINO

Para el correcto funcionamiento del sensor hay que poner una resistencia de 4.7K del pin de Datos y Vcc, Normalmente este sensor viene blindado en un cable largo para aplicaciones donde es necesario sumergirlo en líquidos u otras sustancias. Esta presentación del sensor solo trae 3 terminales o cables de conexión, El pin de Vcc es el cable Rojo, GND es el cable Negro y el Cable de datos puede ser de color Amarillo o Blanco.



29.FIGURA 2.29. ESQUEMA DE CONEXIONADO ENTRE EL DS18B20 Y ARDUINO UNO (PROPIO)

2.6. SISTEMA DE DETECCION DE PH

2.6.1. SENSOR DE PH SEN 0161

30. FIGURA 2.30. SENSOR DE PH SEN 0161 **(EL-KHOURI, 2016)**

El sensor de Potencial de Hidrógeno (pH) es un transductor que permite conocer el pH de una solución, esto lo realiza a través de un método electroquímico que utiliza una membrana de vidrio que separa dos sustancias con diferentes cantidad de, el sensor es un elemento pasivo que genera una pequeña cantidad de corriente de acuerdo al nivel de pH que se encuentre en el medio ambiente. El sensor de pH seleccionado permite una rápida solución para diseños de bajo costo sin que se sacrifique la operatividad del diseño, además de ser un sistema embebido ya que el sensor consta como hemos indicado en el párrafo anterior de una sonda de pH, que es un elemento pasivo que detecta una pequeña corriente eléctrica generada por la actividad de los iones de Hidrógeno. (EL-KHOURI, 2016)

Consta de tres elementos:

- Sensor de pH.
- Placa de conversión a ARDUINO.
- Cable de conexionado entre la placa ARDUINO y el conversor.

El sensor tiene un LED que funciona como el indicador de alimentación, un conector BNC para unirse a la interfaz del sensor PH2.0. Sólo puede conectar el sensor de pH con conector BNC y conectar la interfaz PH2.0 a cualquier entrada analógica del controlador ARDUINO para leer el valor del pH.

Paso para usar el medidor de pH. (EL-KHOURI, 2016)

2.6.2. ESPECIFICACIONES TECNICAS DEL SENSOR DE PH SEN 0161

VOLTAJE DE FUNCIONAMIENTO:	5V DC.
CORRIENTE DE FUNCIONAMIENTO:	
TIEMPO DE RESPUESTA:	≤ 1min
RANGO DE MEDICIÓN	0-14
PRECISIÓN	± 0.1pH (25 ℃)
POTENCIOMETRO	Potenciómetro de ajuste de ganancia
INIDCADORES	Indicador de encendido LED
MEDICIÓN DE LA TEMPERATURA	0-60 ℃
CONECTOR SENSOR	Conector BNC
CONECTOR INTERFAZ PH 2.0	Conector de 3 pines
DIMENSIONES DEL ADAPTADOR:	43mm × 32mm

TABLA 2.2 ESPECIFICACIONES TECNICAS DEL SENSOR DE PH 0161

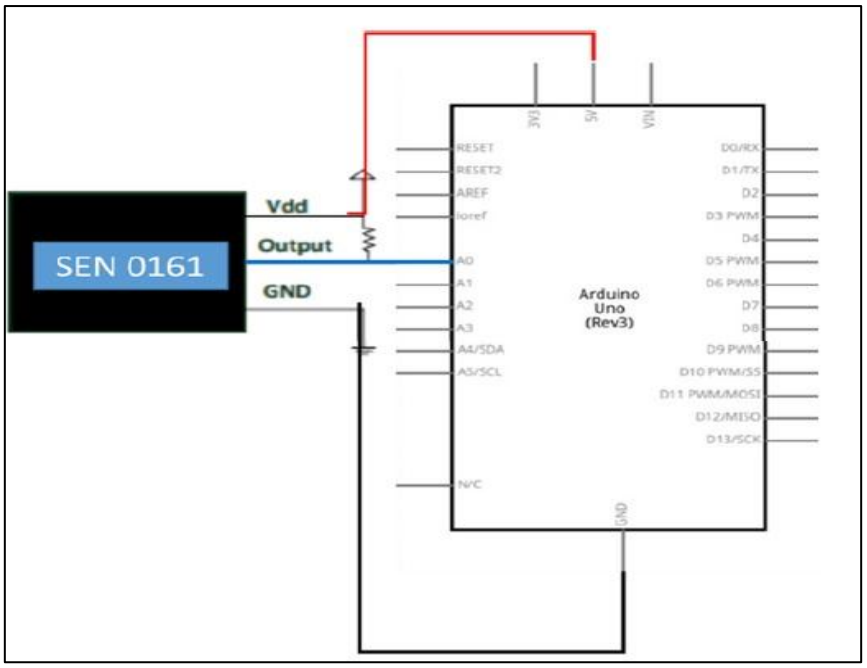
Característica electrodo.

La salida del electrodo del pH es Millivolts, y el valor de pH de la relación se muestra como sigue (25℃):

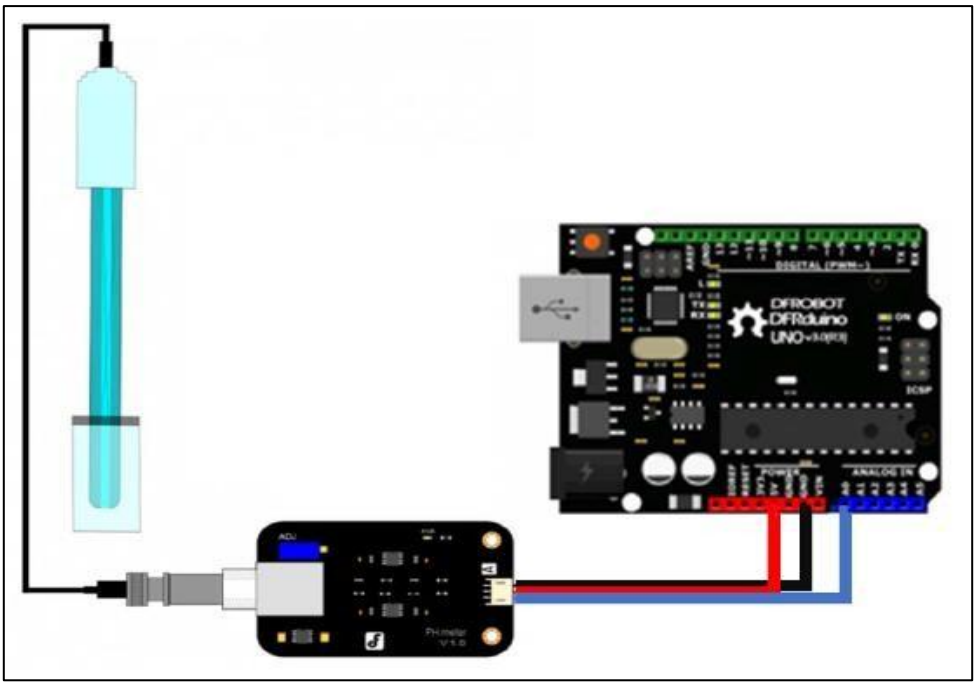
VOLTAGE （mV）	pH value	VOLTAGE （mV）	pH value
414.12	0.00	-414.12	14.00
354.96	1.00	-354.96	13.00
295.80	2.00	-295.80	12.00
236.64	3.00	-236.64	11.00
177.48	4.00	-177.48	10.00
118.32	5.00	-118.32	9.00
59.16	6.00	-59.16	8.00
0.00	7.00	0.00	7.00

TABLA 2.3 SALIDA DEL ELECTRODO DE PH A 25°C

Para lograr el objetivo de monitorear en tiempo real EL PH, en la figura 2.31 y 2.32 se presenta el diagrama a bloques entre el sensor SEN0161 y la placa ARDUINO UNO R3 a través del pin analógico número A0 con una comunicación 1-wire.



31.FIGURA 2.31. CONEXIÓN DEL SENSOR DE PH SEN 0161 CON ARDUINO UNO (PROPIO)



32.FIGURA 2.32. CONEXIÓN DEL SENSOR DE PH SEN 0161 CON ARDUINO UNO (PROPIO)

2.7. SISTEMA DE COMUNICACIONES MOVILES

2.7.1. INTRODUCCIÓN A LOS SISTEMAS DE COMUNICACIONES MÓVILES.

El propósito de un sistema de comunicaciones móvil es, como su nombre indica, prestar servicios de telecomunicaciones entre estaciones móviles y estaciones terrenas fijas, o entre dos estaciones móviles. Existen dos formas de comunicaciones móviles: inalámbrica y celular.

- Comunicación inalámbrica: El radio de acción de esta tecnología es muy limitado. De hecho los equipos móviles y los de transmisión-recepción deben estar situados en zonas geográficas muy cercanas, como por ejemplo, dentro de un mismo edificio.
- Comunicación celular: Tiene una red totalmente definida que incluye protocolos para establecer y despejar llamadas así como rastrear las unidades móviles dentro de áreas geográficas definidas llamadas células, que dan nombre a la tecnología. Dado que los sistemas celulares operan con una potencia más alta que los inalámbricos, el radio de acción de los primeros es mucho más extenso, siendo el tamaño de las células del orden de kilómetros. (Dadateca.unad.edu.co, 2014)

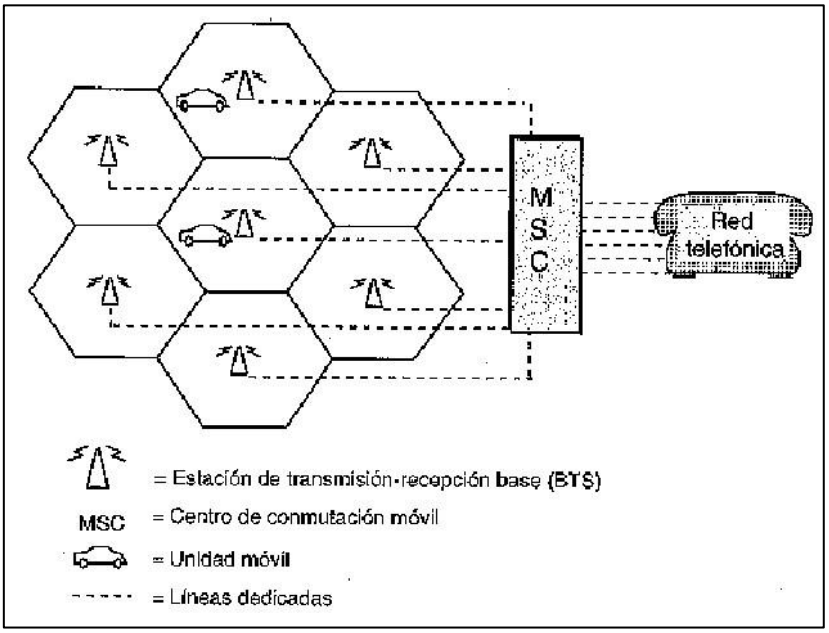
2.7.2. TOPOLOGÍA DE UN SISTEMA CELULAR.

Los componentes principales de un sistema celular son:

- El centro de conmutación móvil (MSC, Mobile Switching Center), que es el centro de control de los sistemas celulares; se encarga de conmutar las llamadas a las células, proporcionar respaldo, conectarse con las redes telefónicas, monitorizar el tráfico para fines de cobro, realizar pruebas y diagnósticos, y realizar labores de administración de la red en general.
- Las células, que son las distintas áreas geográficas en las que se divide el área total que pretende cubrir el sistema.
- La unidad móvil, que es el transmisor-receptor móvil, casi siempre situado en un automóvil, camión, embarcación, etc., y que contiene un módem

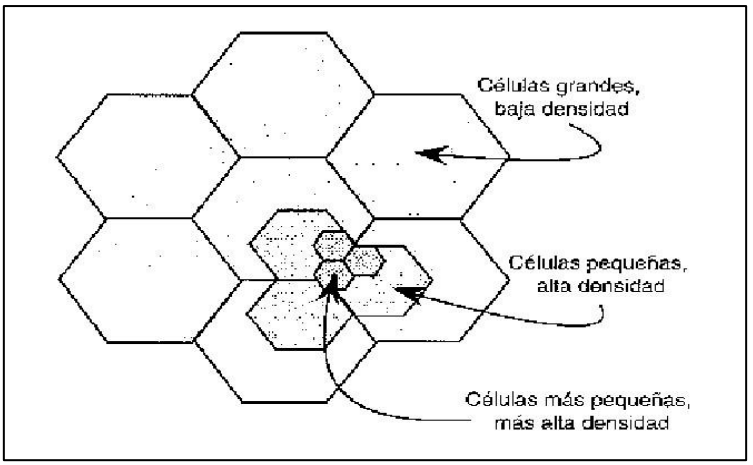
capaz de cambiar de frecuencia que le permite sincronizarse con una frecuencia dada, designada por el MSC.

- La estación de transmisión-recepción base (BTS, Base Transceiver Station). Existe una por cada célula y junto a ésta es la interfaz entre la unidad móvil y el MSC.



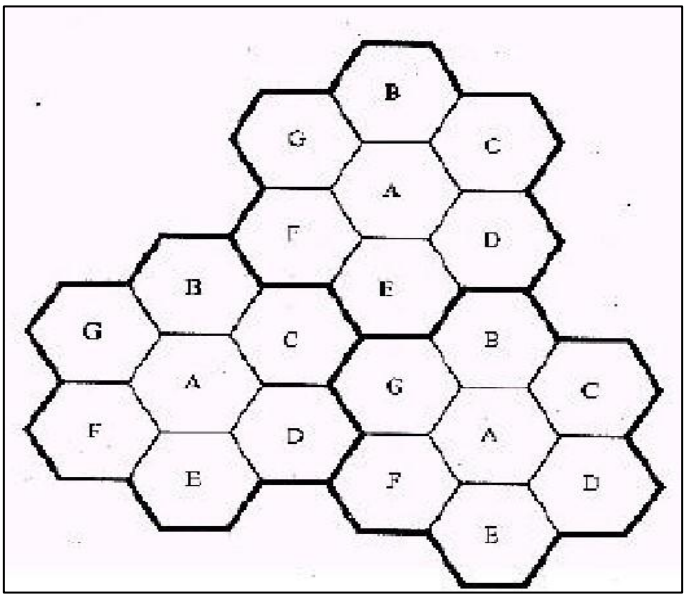
33 FIGURA 2.33. TOPOLOGÍA REPRESENTATIVA DE UN SISTEMA CELULAR.
(Dadateca.unad.edu.co, 2014)

En áreas urbanas muy pobladas, el volumen tan alto de tráfico local puede agotar los canales de radio disponibles. No obstante, es posible aumentar hasta cierto punto la capacidad del sistema reduciendo continuamente el tamaño de las células y la potencia transmitida de las estaciones base. La reducción en el radio de las células permite reutilizar las bandas disponibles en células no contiguas. La estrategia permite al proveedor de portadora celular reducir y aumentar el tamaño de las células para dar cabida al crecimiento o a la reducción de las poblaciones de esta base de suscriptores móviles.



34 FIGURA 2.34. DIVISIÓN DE CÉLULAS. (Dadateca.unad.edu.co, 2014)

Debe hacerse hincapié en que la partición de células requiere un diseño cuidadoso durante el establecimiento inicial del sistema, a fin de minimizar la cantidad de ajustes que es preciso hacer al sistema. Además, si las células son pequeñas se requieren transferencias de control más frecuentes (cuando la unidad móvil pasa de una célula a otra), lo que aumenta el gasto extra de la red. (Dadateca.unad.edu.co, 2014)



35 FIGURA 2.35. EJEMPLO DE REUTILIZACIÓN DE FRECUENCIAS. (Dadateca.unad.edu.co, 2014)

2.7.3. INTERFERENCIAS Y CAPACIDAD DEL SISTEMA

La interferencia es el principal factor que limita el desarrollo de los sistemas celulares. Las fuentes de interferencias incluyen a otras estaciones móviles dentro de la misma celda, o cualquier sistema no celular que de forma inadvertida introduce energía dentro de la banda de frecuencia del sistema celular. Las interferencias en los canales de voz causan el "cross-talk", consistente en que el abonado escucha interferencias de fondo debidas a una transmisión no deseada. Sobre los canales de control, las interferencias conducen a llamadas perdidas o bloqueadas debido a errores en la señalización digital. Las interferencias son más fuertes en las áreas urbanas, debido al mayor ruido de radio frecuencia y al gran número de estaciones base y móviles. Las interferencias son las responsables de formar un cuello de botella en la capacidad y de la mayoría de las llamadas entrecortadas.

Los dos tipos principales de interferencias generadas por sistemas son las interferencias co-canal y las interferencias entre canales adyacentes. Aunque las señales de interferencia se generan frecuentemente dentro del sistema celular, son difíciles de controlar en la práctica (debido a los efectos de propagación aleatoria).

Pero las interferencias más difíciles de controlar son las debidas a otros usuarios de fuera de la banda (de otros sistemas celulares, por ejemplo), que llegan sin avisar debido a los productos de intermodulación intermitentes o a sobrecargas del terminal de otro abonado. En la práctica, los transmisores de portadoras de sistemas celulares de la competencia, son frecuentemente una fuente significativa de interferencias de fuera de banda, dado que la competencia frecuentemente coloca sus estaciones base cerca, para proporcionar una cobertura comparable a sus abonados. (Dadateca.unad.edu.co, 2014)

2.7.4. INTERFERENCIA CO-CANAL Y CAPACIDAD DEL SISTEMA

La reutilización de frecuencias implica que en un área de cobertura dada haya varias celdas que usen el mismo conjunto de frecuencias. Estas celdas son llamadas celdas co-canales, y la interferencia entre las señales de estas celdas se le llama interferencia co-canal. Al contrario que el ruido térmico, que se puede superar incrementando la relación señal ruido ("Signal to Noise Ratio" ó SNR),

la interferencia co-canal no se puede combatir simplemente incrementando la potencia de portadora de un transmisor. Esto es debido a que un incremento en la potencia de portadora de transmisión de una celda, incrementa la interferencia hacia las celdas co-canales vecinas. Para reducir la interferencia co-canal las celdas co-canales deben estar físicamente separadas por una distancia mínima que proporcione el suficiente aislamiento debido a las pérdidas en la propagación. (Dadateca.unad.edu.co, 2014)

2.7.5 INTERFERENCIA ENTRE CANALES ADYACENTES

Entran en este apartado las interferencias procedentes de señales que son adyacentes en frecuencia a la señal deseada. Estas interferencias están producidas por la imperfección de los filtros en los receptores que permiten a las frecuencias cercanas colarse dentro de la banda pasante. El problema puede ser particularmente serio si un usuario de un canal adyacente está transmitiendo en un rango muy próximo al receptor de un abonado, mientras que el receptor está intentando recibir una estación base sobre el canal deseado. A esto se le suele llamar efecto "nearfar", donde un transmisor cercano (que puede ser o no del mismo tipo que el usado en el sistema celular) captura al receptor del abonado. Otra forma de reducir el mismo efecto es cuando un móvil cercano a una estación base transmite sobre un canal cercano a otro que está usando un móvil débil. La estación base puede tener dificultad para discriminar al usuario móvil deseado del otro debido a la proximidad entre los canales.

Este tipo de interferencias se pueden minimizar filtrando cuidadosamente, y con una correcta asignación de frecuencias. Dado que cada celda maneja sólo un conjunto del total de canales, los canales a asignar en cada celda no deben estar próximos en frecuencias. (Dadateca.unad.edu.co, 2014)

2.7.6. CONTROL DE POTENCIA PARA REDUCIR LAS INTERFERENCIAS

En los sistemas celulares de radio, los niveles de potencia transmitida por cada unidad de los abonados, están bajo un control constante por las estaciones base servidoras. Esto se hace para asegurar que cada móvil transmite la potencia más baja necesaria y así reducir las interferencias entre canales. (Dadateca.unad.edu.co, 2014)

2.7.7. TIPOS DE SISTEMAS CELULARES E IMPACTO EN EL MERCADO

A. Primeros sistemas celulares en kHz y km. [LEE89]

	AMPS	*TACS*	*NMT*	*T(450C)*
Estación base	870–890	935–960	463–467.5	461.3–465.74
Estación móvil	825–845	870–915	453–457.5	451.3–455.74
Espaciado	45	45	10	10
Radio de cobertura	2–20	2–20	1.8–40	5–30
Modulación	FM	FM	FM	FM

B. Crecimiento mundial de suscriptores (millones de suscriptores) 1994–1995 [PCSC95]

	6/95	*12/94*
Europa	18.5	14.7
Asia-Pacífico	15.6	11.1
Norteamérica	28.2	26.0
América del Sur/Central	3.0	2.4
Medio Oriente	0.5	0.4
África	0.6	0.3
Total	66.4	54.9

C. Uso de la tecnología (millones de suscriptores): 1994–1995

	6/95	*12/94*
Analógica		
AMPS	35.5	32.4
TACS	12.3	9.5
NMT-450	1.4	1.4
NMT-900	3.0	2.7
NTT	2.3	1.9
Otras	0.9	1.0
Subtotal	55.4	49.3
Digital		
GSM	7.4	4.6
PDC	1.5	0.5
DCS-1800	0.6	0.4
TDMA	1.5	1.0
Subtotal	11.0	6.0
Total	66.4	54.9

36 FIGURA. 2.36. RESUMEN DE SISTEMAS CELULARES. (Dadateca.unad.edu.co, 2014)

Estos sistemas son incompatibles entre sí, lo cual dio lugar a plantearse la implantación de un sistema celular a nivel mundial. He aquí la razón de ser del modelo GSM.

2.8. GSM

2.8.1 INICIOS

Los primeros trabajos con GSM los inició en 1982 un grupo dentro del Instituto Europeo de Normas de Comunicaciones (ETSI, European Telecommunications Standards Institute). Originalmente, este organismo se llamaba Groupe Sociale Mobile, lo que dio pie al acrónimo GSM.

El objetivo de este proyecto era poner fin a la incompatibilidad de sistemas en el área de las comunicaciones móviles y crear una estructura de sistemas de comunicaciones a nivel europeo.

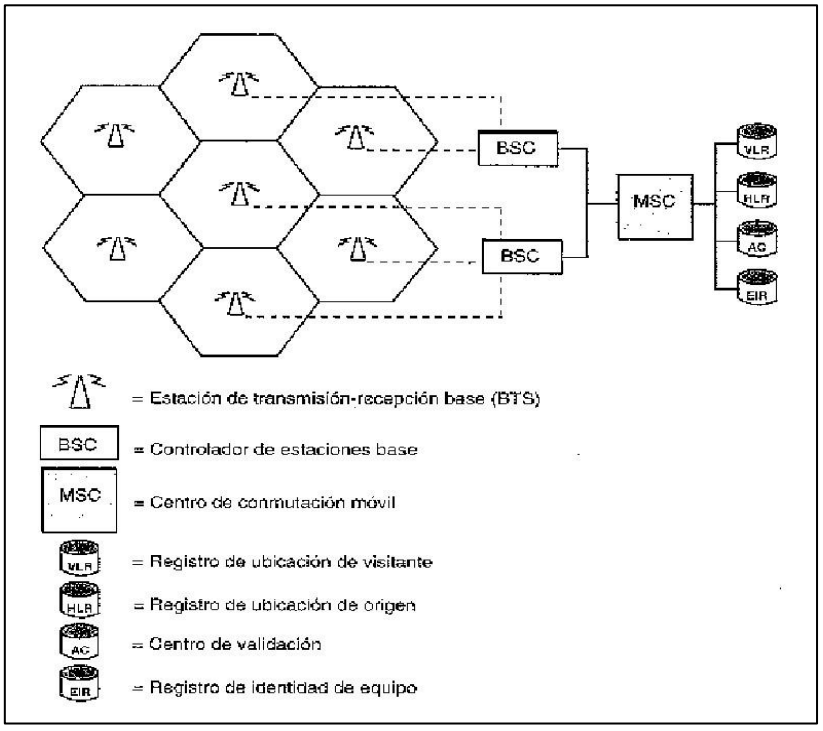
GSM se diseñó para incluir una amplia variedad de servicios que incluyen transmisiones de voz y servicios de manejo de mensajes entre unidades móviles o cualquier otra unidad portátil. (Dadateca.unad.edu.co, 2014)

2.8.2 COMPONENTES DE GSM

Los componentes principales GSM son:

- El centro de conmutación móvil (MSC, Mobile Switching Center), es el corazón de todo sistema GSM y se encarga de establecer, gestionar y despejar conexiones, así como de enrutar las llamadas a la célula correcta. El MSC proporciona la interfaz con el sistema telefónico y presta servicios de determinación de cargos y contabilidad.
- La célula, cuyo tamaño es de aproximadamente 35 km.
- La unidad móvil (MS, Mobile Station).
- El controlador de estaciones base (BSC, Base Station Controller). Es un elemento nuevo introducido por GSM. Se encarga de las operaciones de transferencia de control de las llamadas y también de controlar las señales de potencia entre las BTS y las MS, con lo cual releva al centro de conmutación de varias tareas.
- La estación de transmisión-recepción base (BTS, Base Transceiver Station).Establece la interfaz a la unidad móvil. Está bajo el control del BSC.

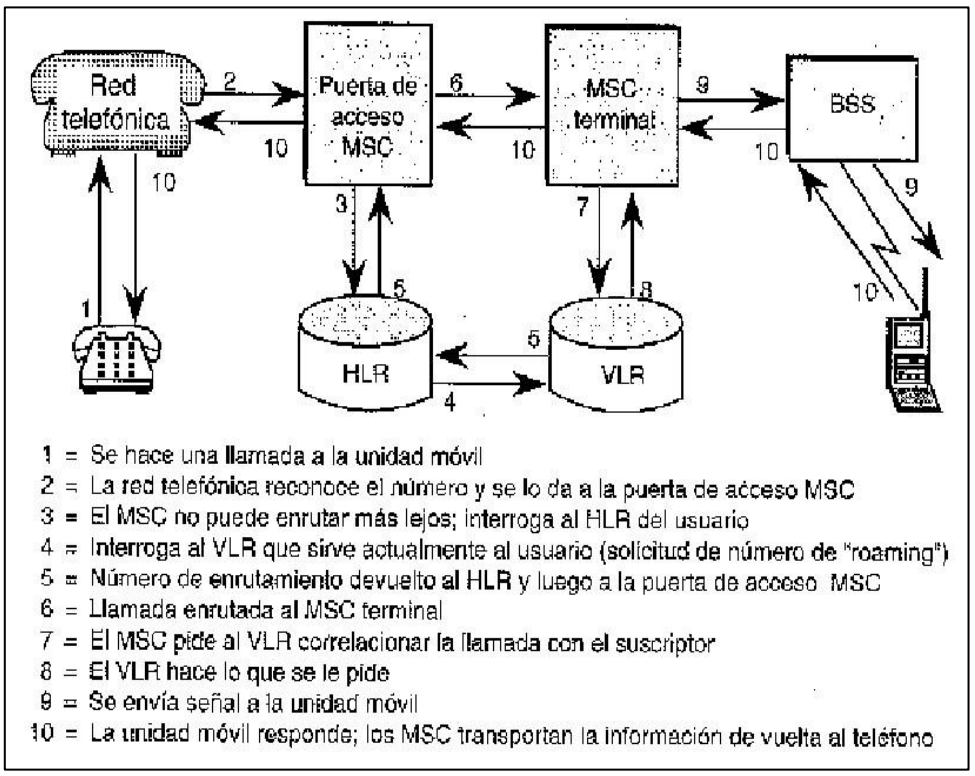
- La HLR (Home Location Register) es una base de datos que proporciona información sobre el usuario, su base de suscripción de origen y los servicios suplementarios que se le proveen.
- El VLR (Visitor Location Register) es también una base de datos que contiene información sobre la situación de encendido/apagado de las estaciones móviles y si se han activado o desactivado cualesquiera de los servicios suplementarios.
- El centro de validación (AC o AUC, Authentication Center) que sirve para proteger a cada suscriptor contra un acceso no autorizado o contra el uso de un número de suscripción por personas no autorizadas; opera en relación estrecha con el HLR.
- El registro de identidad del equipo (EIR, Equipment Identity Register) que sirve para registrar el tipo de equipo que existe en la estación móvil y también puede desempeñar funciones de seguridad como bloqueo de llamadas que se ha determinado que emanan de estaciones móviles robadas, así como evitar que ciertas estaciones que no han sido aprobadas por el proveedor de la res usen ésta. (Dadateca.unad.edu.co, 2014)



37 FIGURA 2.37. ESQUEMA DE COMPONENTES GSM. (Dadateca.unad.edu.co, 2014)

2.8.3 ENRUTAMIENTO DE LLAMADAS

En la figura 2.38 se muestra un ejemplo de enrutamiento de llamadas GSM. En el paso 1, un usuario de teléfono llama a la unidad móvil a través de la red telefónica pública. La llamada se enruta a un MSC de puerta (paso 2), el cual examina los dígitos marcados y determina que no puede enrutar la llamada más lejos; por tanto, en el paso 3, interroga el registro de ubicación de origen (HLR) del usuario llamado a través del SS7 TCAP (transation capabilities application part). El HLR interroga el registro de ubicación de visitante (VLR) que actualmente está dando servicio al usuario (paso 4). En el paso 5, el VLR devuelve un número de enrutamiento al HLR, que lo devuelve al MSC de puerta. Con base en este número de enrutamiento, el MSC de puerta enruta la llamada al MSC terminal (paso 6). El MSC terminal consulta entonces el VLR para comparar la llamada entrante con la identidad del suscriptor receptor (pasos 7 y 8). En el paso 9, la BSS recibe una solicitud de notificación del MSC terminal y envía una señal de notificación. Cuando la señal de usuario regresa, la llamada se completa (paso 10). (Dadateca.unad.edu.co, 2014)

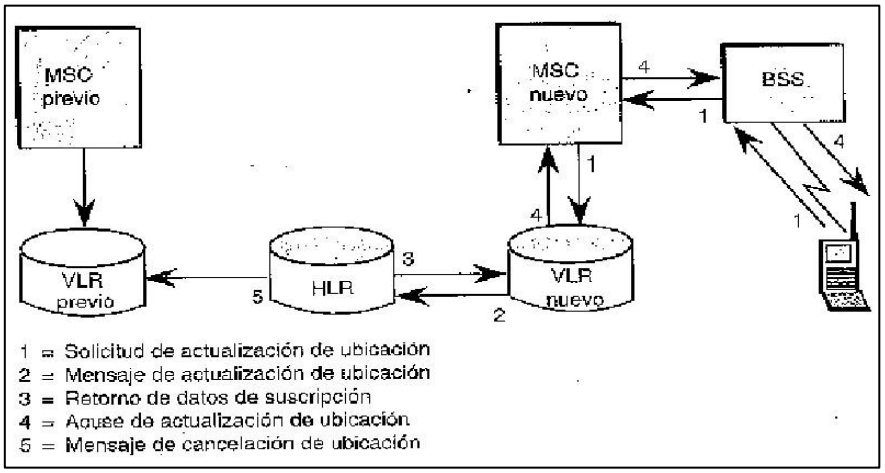


38 FIGURA 2.38. EJEMPLO DE GESTIÓN DE LLAMADAS GSM. (Dadateca.unad.edu.co, 2014)

2.8.4 ACTUALIZACIÓN DE UBICACIÓN

La figura 2.39 muestra un ejemplo de cómo un suscriptor puede vagar de una célula a otra y de cómo el sistema sigue la posta de dicho suscriptor. Cuando una estación móvil cruza una frontera de una célula, la unidad móvil envía automáticamente su solicitud de actualización de ubicación (que también contiene su identificación) a la BSS. El mensaje se enruta al MSC de la nueva célula, que examina su VLR (VLR nueva en la figura 2.9). Si la VLR nueva no tiene información acerca de la identidad del mensaje para este usuario (porque el usuario llegó hace poco a esta área), envía un mensaje de solicitud de actualización de ubicación al registro de ubicación de origen del usuario (suceso 2). Este mensaje incluya la identidad del usuario así como la identidad del VLR que está enviando el mensaje. En el suceso 3, el HLR almacena la nueva ubicación que está enviando el mensaje. En el suceso 3, el HLR almacena la nueva ubicación del suscriptor como VLR nuevo y luego carga línea abajo la base de datos de suscripción del usuario en el nuevo VLR. Al recibir esta información, el nuevo VLR envía el acuse de recibo de la actualización de ubicación a través del nuevo MSC a la BSS y de vuelta al usuario móvil originador (suceso 4). Por último, en el suceso 5, el HLR envía un mensaje de cancelación de ubicación al VLR viejo para borrar los datos del suscriptor de su base de datos.

Importante, sólo un VLR a la vez debe conocer al suscriptor móvil. En este ejemplo, cuando el suscriptor se ha movido a otra área (otra célula), ha sido necesario actualizar el VLR.



39 FIGURA 2.39. ACTUALIZACIÓN DE UBICACIÓN. (Dadateca.unad.edu.co, 2014)

Es evidente que el HLR es el maestro de las bases de datos de suscriptores y por tanto coordina los cambios a los VLR y MSC conforme el suscriptor se mueve de una célula a otra. (Dadateca.unad.edu.co, 2014)

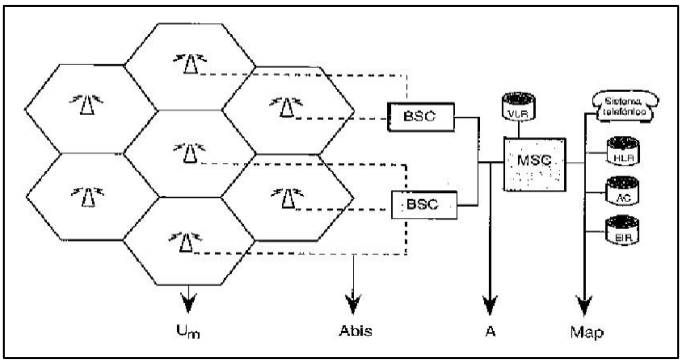
2.8.5. GSM 900/DCS 1800: CIMIENTOS DE PCS 1900 (TDMA)

En Norteamérica, varios proveedores de servicios han escogido al PCS 1900 de acceso múltiple por división en el tiempo (TDMA, Time Division Multiple Acces) como tecnología de segunda generación para las redes móviles-inalámbricas. PCS 1900 (TDMA) es muy similar a GSM 900/DCS 1800, y utiliza el mismo protocolo Um; opera en el espectro de 1900 MHz.

Estos sistemas móviles-inalámbricos de segunda generación emplean técnicas similares para establecer los canales físicos y lógicos en la interfaz de radio. Las diferencias principales son las frecuencias que se usa para los canales físicos. Los canales lógicos (las slots de tiempo) son muy similares y se clasifican como canales de tráfico (TCH) o canales de control (CCH). Los canales físicos se designan con n, donde n es el ARFCN (Absolute Radio Frequency Channel Number, número absoluto de canal radiofrecuencia). El sistema GSM 900 emplea dos bandas de 25 MHz para el enlace ascendente y el enlace descendente. Dentro de ese espectro se asignan canales de 200 KHz. El enlace ascendente y el descendente están separados por un espaciado de 45 MHz. El ARFCN varía entre q y 124.

La asignación de los canales de 100 KHz varía y depende de los patrones de tráfico y del tamaño de célula del sistema. El sistema DCS 1800 usa dos bandas de 75 MHz para el enlace ascendente y el descendente. Al igual que en GSM 900, se asignan canales de 200 KHz dentro de esas bandas. El enlace ascendente y el descendente están separados por un espaciado de 95 MHz. El ARFCN varía entre 512 y 885. En PCS 1900 (TDMA), el sistema usa dos bandas de 60 MHz para el enlace ascendente y el enlace descendente. Al igual que los otros sistemas, PCS 1900 usa canales de 200 KHz con el enlace ascendente y el descendente separados por un espaciado de 80 MHz. (Dadateca.unad.edu.co, 2014)

2.8.6. INTERFACES GSM



40 FIGURA 2.40. LAS INTERFACES GSM. (Dadateca.unad.edu.co, 2014)

GSM se diseñó de modo que permitiera la división en particiones funcionales. Dichas particiones tienen sus fronteras en las diferentes interfaces que la componen. Estas son las siguientes:

- La interfaz **A**. Un lado de la interfaz se ocupa de las operaciones de MSC, HLR y VLR, y el otro lado de ella se encarga de las operaciones de BSC y de radio.
- Una segunda interfaz llamada **Abis**, defines las operaciones entre el BSC y la BTS; se basa en un enlace de transmisión PCM-30 de 2 Mbit/s y LAPD.
- La interfaz de aplicación móvil, **MAP (Mobile Application Part)** define las operaciones entre el MSC y la red telefónica, así como el MSC, el HLR, el VLR y el EIR. MAP se implementa encima de SS7.
- La interfaz de radio **Um**, a la cual dedicamos un completo apartado debido a su transcendental importancia. (Dadateca.unad.edu.co, 2014)

2.9. SMS

2.9.1. DEFINICIÓN

Servicio de mensajes cortos. Es un sistema para enviar y recibir mensajes de texto para y desde teléfonos móviles. El texto puede estar compuesto de palabras o números o una combinación alfanumérica. SMS fue creado como una parte del estandar GSM fase 1. El primer mensaje corto, se cree que fue enviado en Diciembre de 1992 desde un ordenador personal (PC) a un teléfono móvil a través de la red GSM Vodafone del Reino Unido. Cada mensaje puede tener

hasta 160 caracteres cuando se usa el alfabeto latino, y 70 caracteres si se usa otro alfabeto como el árabe o el chino. (Dadateca.unad.edu.co, 2014)

2.9.2. CARACTERÍSTICAS

Hay varias características únicas del servicio de mensajes cortos (SMS), según lo definido dentro del estándar digital de telefonía móvil GSM, un mensaje corto puede tener una longitud de hasta 160 caracteres. Esos 160 caracteres pueden ser palabras, números o una combinación alfanumérica. Los mensajes cortos basados en No-texto (por ejemplo, en formato binario) también se utilizan. Los mensajes cortos no se envían directamente del remitente al receptor, sino que se envían a través de un centro de SMS. Cada red de telefonía móvil que utiliza SMS tiene uno o más centros de mensajería para manejar los mensajes cortos. El servicio de mensajes cortos se caracteriza por la confirmación de mensaje de salida. Esto significa que el usuario que envía el mensaje, recibe posteriormente otro mensaje notificándole si su mensaje ha sido enviado o no. Los mensajes cortos se pueden enviar y recibir simultáneamente a la voz, datos y llamadas del fax. Esto es posible porque mientras que la voz, los datos y las llamadas del fax asumen el control de un canal de radio dedicado durante la llamada, los mensajes cortos viajan sobre un canal dedicado a señalización independiente de los de tráfico. Hay formas de enviar múltiples mensajes cortos:

- La concatenación SMS (que encadena varios mensajes cortos juntos).[3]
- La compresión de SMS (que consigue más de 160 caracteres de información dentro de un solo mensaje corto).

Para utilizar el servicio de mensajes cortos, los usuarios necesitan la suscripción y el hardware específico:

- Una suscripción a una red de telefonía móvil que soporte SMS.
- Un teléfono móvil que soporte SMS.
- Un destino para enviar o recibir el mensaje, ya sea una máquina de fax, un PC, un terminal móvil o un buzón de e-mail. (Dadateca.unad.edu.co, 2014)

2.10. TECNOLOGIA GPRS

2.10.1. INTRODUCCION

GPRS es una nueva tecnología que comparte el rango de frecuencias de la red GSM utilizando una transmisión de datos por medio de paquetes. La conmutación de paquetes es un procedimiento más adecuado para transmitir datos, hasta ahora los datos se habían transmitido mediante conmutación de circuitos, procedimiento más adecuado para la transmisión de voz. La tecnología GPRS, o generación 2.5, representa un paso más hacia los sistemas inalámbricos de Tercera Generación o UMTS. Su principal base radica en la posibilidad de disponer de un terminal permanentemente conectado, tarificando únicamente por el volumen de datos transferidos (enviados y recibidos) y no por el tiempo de conexión como hemos podido observar en un punto anterior.

Obtiene mayor velocidad y mejor eficiencia de la red. Tradicionalmente la transmisión de datos inalámbrica se ha venido realizando utilizando un canal dedicado a GSM a una velocidad máxima de 9.6 Kbps, con el GPRS no sólo la velocidad de transmisión de datos se ve aumentada hasta un mínimo 40 Kbps y un máximo de 115 Kbps por comunicación, sino que además la tecnología utilizada permite compartir cada canal por varios usuarios, mejorando así la eficiencia en la utilización de los recursos de red. La tecnología GPRS permite proporcionar servicios de transmisión de datos de una forma más eficiente a como se venía haciendo hasta el momento.

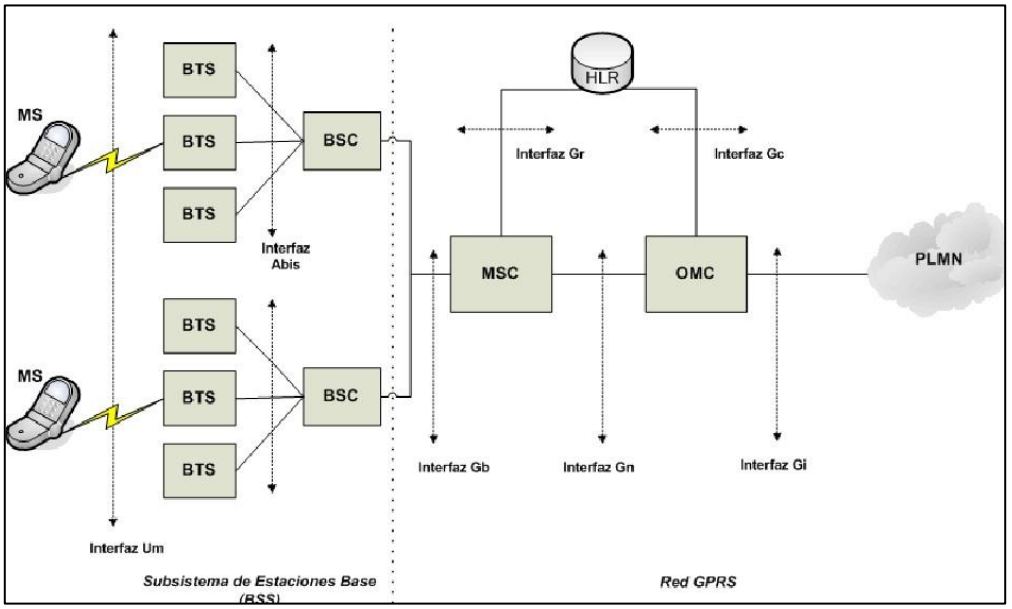
GPRS es una evolución no traumática de la actual red GSM: no conlleva grandes inversiones y reutiliza parte de las infraestructuras actuales de GSM. Por este motivo, GPRS tiene, desde sus inicios, la misma cobertura que la actual red GSM.

GPRS (Global Packet Radio Service) es una tecnología que subsana las deficiencias de GSM.

La Figura 2.41 ilustra la arquitectura del sistema GPRS. Comparado con el sistema GSM, GPRS introduce 2 nuevos elementos, (que se encuentran sombreados en dicha figura) para crear un modo de transferencia de paquetes end to end. (Yeferson Bedoya Giraldo, 2013)

Se proveen dos servicios:

- Punto a Punto (PTP).
- Punto a Multipunto (PTM).



41 FIGURA 2.41. ARQUITECTURA DEL SISTEMA GPRS. (Yeferson Bedoya Giraldo, 2013)

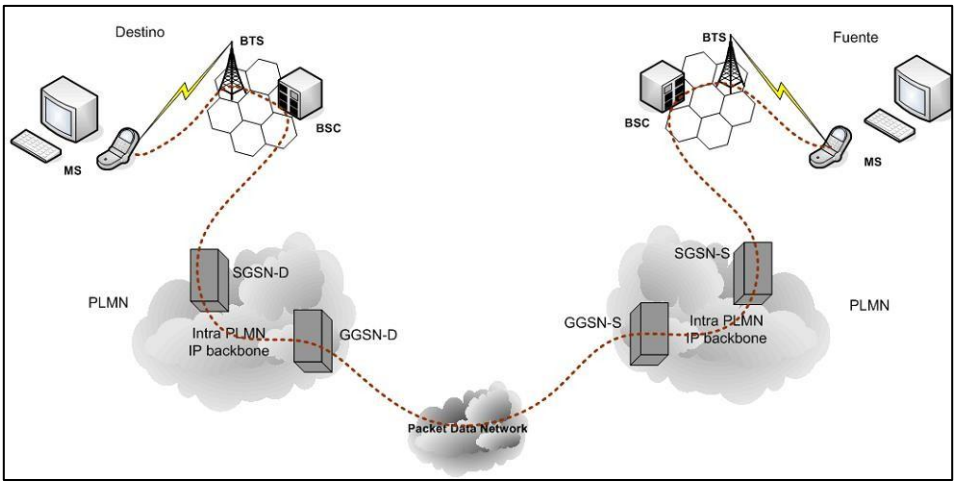
Independientemente del ruteo de paquetes y la transferencia dentro de la red móvil pública terrestre, dentro es soportado un nuevo nodo de red lógico llamado el Nodo de Soporte GPRS. El nodo de soporte de salida GPRS actúa como una interfaz lógica hacia las redes de paquetes de datos externas. El nodo de soporte de servicio GPRS es responsable por la entrega de paquetes a las MS's dentro de su área de servicio.

Dentro de la red GPRS, las unidades de protocolo de datos son encapsuladas en el GSN origen y des encapsuladas en el GSN destino. Entre los GSNs el Protocolo de Internet es utilizado como el Backbone para transferir PDUs. Todos los datos GPRS relativos al usuario necesarios para que el SGSN desempeñe sus funciones de ruteo y transferencia de datos son almacenados dentro del HLR.

La Figura 2.42 muestra un ejemplo simple de ruteo en una transmisión. El SGSN de la estación móvil fuente (SGSN-S), encapsula los paquetes transmitidos por la MS y los envía al correspondiente GGSN.

Basándose en la exanimación de la dirección destino, los paquetes son entonces ruteados al GGSN-D a través de la red de paquetes de datos. El GGSN-D

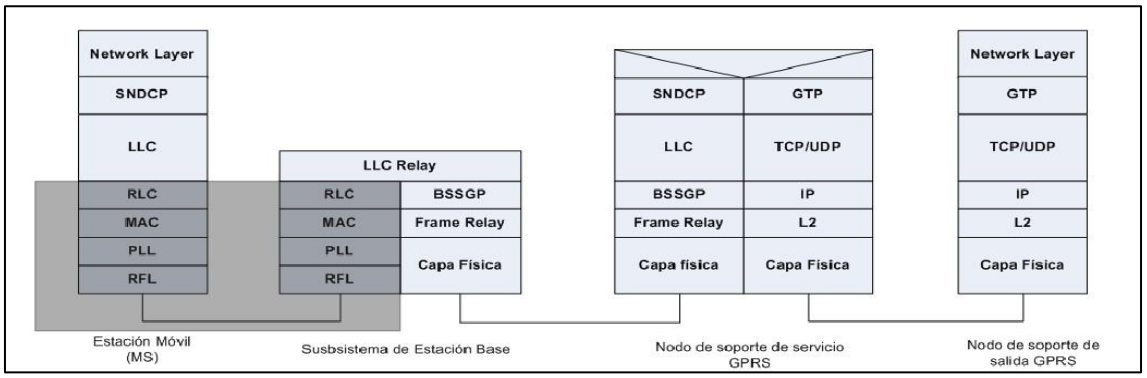
chequea el contexto del ruteo asociado con la dirección y destino, determina el SGSN sirviendo al destino (SGSN-D) y la información relevante al tune. Cada paquete es entonces encapsulado y reenviado al SGSN-D, que lo entrega finalmente a la MS destino. (Yeferson Bedoya Giraldo, 2013)



42 FIGURA 2.42. EJEMPLO DE RUTEO EN UNA RED GPRS. (Yeferson Bedoya Giraldo, 2013)

2.10.2. ARQUITECTURA DEL PROTOCOLO

La Figura 2.43 muestra el plano de transmisión propuesto hasta la capa de red de acuerdo al modelo de referencia OSI. Por encima de la capa de red se pueden utilizar diversos protocolos o estándares pero dicha selección se encuentra fuera del alcance de la especificación GPRS. Bajo el protocolo TCP/UDP y el IPson utilizados los protocolos de la capa de red del Backbonede la red GPRS. Los protocolos basados en Ethernet, ISDN y ATM pueden ser utilizados bajo IP dependiendo de la arquitectura de red del operador.



43 FIGURA 2.43. PLANO DE TRANSMISIÓN GPRS. (Yeferson Bedoya Giraldo, 2013)

Entre el SGSN y la MS, el protocolo SNDC mapea las características del protocolo a nivel de red dentro del LLC (Logical Link Control) y provee funcionalidades tales como: el multiplex de los mensajes de la capa de red dentro de una conexión virtual lógica, la encriptación, la segmentación y la compresión. Las radiocomunicaciones entre una MS y la red GPRS se encuentran indicadas por el área sombreada en la Figura 2.43, y cubren las funciones de las capas físicas y de enlace de datos. (Yeferson Bedoya Giraldo, 2013)

2.11. SIM 900

2.11.1. INTRODUCCION

GSM / GPRS RS232 Módem fabricado por SIMCOM; SIM900 QUAD-BAND de GSM / GPRS, funciona en las frecuencias de 850 MHz, 900 MHz, 1800 MHz y 1900 MHz. Es de tamaño compacto y fácil de usar como plug módem GSM. El módem está diseñado con RS232 Nivel circuitería de convertidor, que le permite conectarse directamente puerto serie del PC . La velocidad de transmisión puede ser configurable a partir 9600-115200 a través de comandos AT. Inicialmente módem está en modo automático de baudios.

Este RS232 GSM / GPRS Módem guarda la TCP / IP interna para que pueda conectar con Internet a través de GPRS. Es adecuado para SMS así como la aplicación de transferencia de datos en la interfaz M2M.

El módem se necesita sólo 3 hilos (Tx, Rx, GND), excepto la fuente de alimentación para la interfaz con microcontrolador / PC Host. El construido en el regulador de voltaje de caída baja lineal permite la conexión de ancho gama de la fuente de alimentación no regulada (4.2V -13V). Sí, 5 V está entra al modem, podrá para enviar y leer SMS, conectarse a Internet a través de GPRS a través de simples comandos AT. (RHYDOLABZ.COM, 2011)

2.11.2. CARACTERISTICAS

- Producto de alta calidad.
- Quad-Band GSM / GPRS 850/900/1800/1900 MHz.
- Construido en RS232 convertidor de nivel (MAX3232).
- Velocidad de transmisión configurable.

- Conector SMA con el GSM Tipo L Antena.
- Bandeja de la tarjeta SIM.
- Led de estado de red.
- Incorporacion de protocolos TCP / IP Potente para la transferencia de datos de Internet a través de GPRS.
- Conector de interfaz de audio.
- La mayoría de estado y Control Pins están disponible en Conector.
- La temperatura normal de funcionamiento: -20 ° C a 55 ° C.
- Voltaje de entrada: 5V-12V DC.

2.11.3. ESPECIFICACIONES

- Quad-Band 850/900/1800/1900 MHz.
- GPRS multi-slot clase 10/8.
- GPRS estación móvil de clase B.
- Cumple con la norma GSM fase 2/2 +.
    - clase 4 (2 W @ 850/900 MHz).
    - Clase 1 (1 W @ 1800 / 1900MHz).
- Dimensiones: 24 * 24 * 3 mm.
- Peso: 3,4 g.
- Control a través de comandos AT (GSM 07.07, 07.05 y SIMCOM mejorado Comandos AT).
- Bajo consumo de energía: 1.0mA (modo descanso).
- Temperatura de funcionamiento: -40 ° C a + 85 ° C.
- Especificaciones para fax.
    - Grupo 3, clase 1.
- Las especificaciones para datos
    - GPRS clase 10: máx. 85,6 kbps (descendente).
    - Soporte PBCCH.
    - Esquemas de codificación CS 1, 2, 3, 4.
    - CSD subida 14,4 kbps.
    - USSD.
    - El modo no transparente.
    - PPP-pila.
- Especificaciones para SMS vía GSM / GPRS.

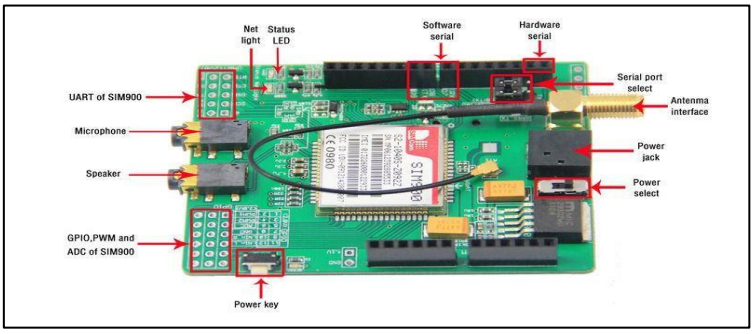
- Punto a punto MO y MT.
- Difusión celular SMS.
- El modo de texto y PDU.

- Las características del software.
  - 0710 protocolo MUX.
  - TCP incrustado / protocolo UDP.
  - FTP / HTTP.
- Firmware especial.
  - MMS.
  - Java.
  - Embedded AT.
- Especificaciones para la voz.
  - Tricodec.
    - La mitad de la velocidad (HR).
    - Tasa completa (FR).
    - Enhanced Full Rate (EFR)
  - Operación manos libres.
  - (Supresión del eco).
  - AMR
    - Media velocidad (HR).
    - Tasa completa (FR).
- Interfaces.
  - Pines de interfaz de audio analógicas a 2 mm Pitch RMC.
  - Interfaz en serie RS232.
  - Conector de antena SMA.
  - Pines de alimentación de CC a 2 mm Pitch RMC.
- Compatibilidad.
  - Interfaz de comando AT celular. (RHYDOLABZ.COM, 2011)

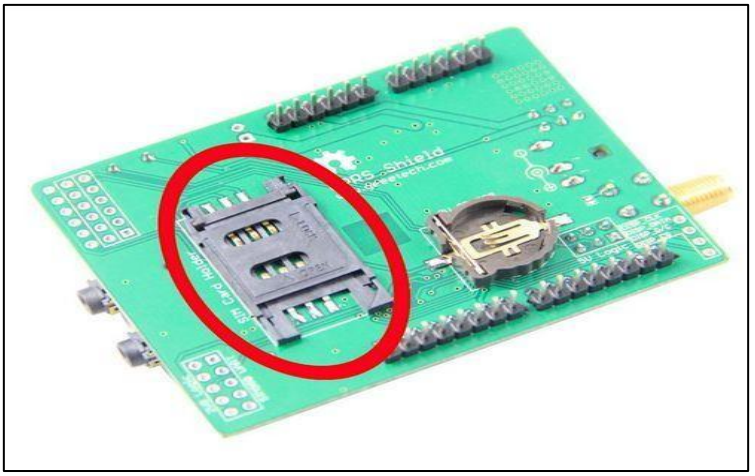
2.11.4. DIAGRAMA DE TARJETA

- Power selector.- seleccionar la fuente de alimentación para el escudo GPRS (alimentación externa o 5V)
- Power jack. - conectado a la fuente de alimentación de 4,8 a 5 V CC externa.

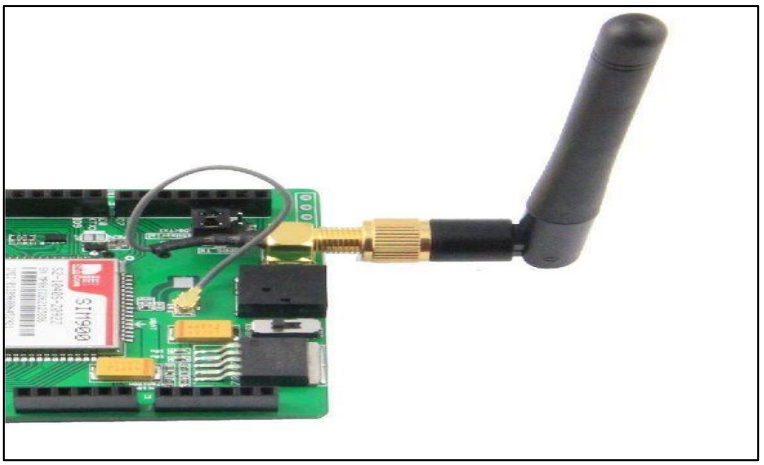
- Antenna interface- conectado a la antena externa.
- Serial port select - Puede seleccionar el software de puerto serie o puerto de serie del hardware para conectarse a GPRS Escudo.
- Hardware Serial- D0 / D1 de Arduino
- Software serial - D7 / D8 de Arduino
- Status LED- dirá si el poder de SIM900 está en
- Net light- dirá el estado SIM900 sobre la vinculación a la red
- UART del SIM900 - pines UART ruptura de SIM900
- Microphone- para responder a la llamada de teléfono
- Speaker - para responder a la llamada de teléfono
- GPIO, PWM y ADC de SIM900 - GPIO, pines PWM y ADC ruptura de SIM900
- Power key- el poder arriba y hacia abajo para SIM900. (GEEETECH.COM, 2014)



44 FIGURA 2.44. TARJETA SIM900 DIAGRAMA DE TARJETA. (GEEETECH.COM, 2014)



45 FIGURA 2.45. BANDEJA DE TARJETA SIM. (GEEETECH.COM, 2014)

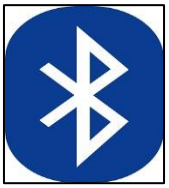


46 FIGURA 2.46. ANTENA DE LA SIM 900. (GEEETECH.COM, 2014)

2.12. BLUETOOTH

Bluetooth es una especificación industrial para Redes Inalámbricas de Área Personal (WPAN) que posibilita la transmisión de voz y datos entre diferentes dispositivos mediante un enlace por radiofrecuencia en la banda ISM de los 2.4 GHz. Los principales objetivos que se pretenden conseguir con esta norma son:

- Facilitar las comunicaciones entre equipos móviles.
- Eliminar los cables y conectores entre estos.
- Ofrecer la posibilidad de crear pequeñas redes inalámbricas y facilitar la sincronización de datos entre equipos personales. (RODRIGUEZ, 2017)



47. FIGURA 2.47. LOGO DE LA SEÑAL BLUETOOTH. **(RODRIGUEZ, 2017)**

Los dispositivos que con mayor frecuencia utilizan esta tecnología pertenecen a sector de las telecomunicaciones y la informática personal, como PDA, teléfonos móviles, computadoras portátiles, ordenadores personales, impresoras o cámaras digitales.

2.12.1 HISTORIA DE BLUETOOTH

El nombre de bluetooth tiene un origen de procedencia nórdica, concretamente del rey de Noruega y Dinamarca llamado Harald Blåtand, cuya traducción literal al inglés sería la de *Harold Bluetooth.* Este noble personaje fue conocido por ser un buenísimo comunicador, el cual hizo la gran gesta de unir las diferentes tribus noruegas, suecas y danesas. (RODRIGUEZ, 2017)

2.12.2. PRINCIPIOS DEL BLUETOOTH

La primera empresa que creó un equipo de trabajo para investigar sobre sistemas de comunicación entre dispositivos, fue Ericsson, que allá por el año 1994 comenzó la investigación de una nueva interfaz de bajo consumo y coste, destinada al envío y recepción de datos entre teléfonos móviles y otros dispositivos.

No fue hasta el año 1999, cuando se creó el SIG de Bluetooth *(Special Interest Group),* el cual consistía en la unión de diferentes empresas, entre las cuales se encontraban en un primer momento Ericsson, Intel, Nokia, Toshiba e IBM. A estas empresas fundadoras del grupo, tan solo unos meses después se les unieron otras empresas de la tecnología tan importantes como Microsoft, 3COM, Motorola **y** Lucent**.** (RODRIGUEZ, 2017)

2.12.3. VERSIONES DE BLUETOOTH

BLUETOOTH V1.0 Y V1.0B

Los primeros emisores receptores de bluetooth, fueron el v1.0 y v1.0B**,** los cuales ya están prácticamente obsoletos, y dieron muchísimos problemas a los fabricantes de teléfonos para la interactuación entre dispositivos de diferentes compañías, asimismo, tenían el gran defecto que en cada transmisión de datos se enviaba nuestra dirección privada de dispositivo bluetooth, perdiendo así el anonimato que nos pudiese brindar este tipo de conexión inalámbrica. (RODRIGUEZ, 2017)

BLUETOOTH V1.1

- Usa el estándar **IEEE 802.15.1-20022**
- Corregidos errores de las versiones anteriores.

- Canales no encriptados añadidos y soportados.
- Añadido el indicador de la señal o también denominado *(RSSI)*

BLUETOOTH V1.2

- Compatibilidad con usb **1.1.**
- Mejora la velocidad de conexión y transferencia de datos.
- Añadida la función de detección de otros dispositivos bluetooth en el radio de actuación.
- Notables mejoras en la calidad del audio.
- **H**ost **C**ontroller **I**nterface *(HCI)*
- Nuevo protocolo estándar IEEE 802.15.1-2005.
- Añadido control de flujo y modos de retransmisión L2CAP.

BLUETOOTH V2.0 + EDR

La mejora implementada en esta nueva versión, hace referencia a la opción del propio fabricante del dispositivo de incorporar la EDR *(Enhanced Data Rate),* esto no viene a significar que todos los dispositivos **2.0** vengan con este sistema de transmisión de datos a mayor velocidad, ya que como he comentado es de carácter opcional. Su transferencia máxima de datos es de 3Mb/s. aunque su tasa real máxima sea la de 2.1Mb/s. está versión mantiene la compatibilidad con la versión anterior de la interfaz bluetooth.

BLUETOOTH V2.1 + EDR

Las mejoras de esta nueva versión, son mejoras sustanciales siempre mirando hacia la seguridad de nuestros datos, así de esta manera se ha añadido **S**ecure **S**imple Pairing (SSP**)**, lo que permite un mejor filtrado de nuestros datos y una seguridad superior a la de la versión anterior. A su vez, se ha mejorado notablemente el consumo de energía, gracias a la nueva tecnología oler subrating.

BLUETOOTH V3.0 + HS

Este nuevo modelo de la interfaz, fue lanzado en abril de 2009, y su mayor logro es el aumento de la velocidad de transmisión de datos hasta los 24Mb/**s**., además de incluir una nueva característica la cual hace uso del wifi para el envío y recepción de grandes paquetes de datos, usando el estándar 802.11 de alta velocidad, esta nueva característica es denominada Alternativa MAC / PHY

BLUETOOTH V4.0

Esta versión es la más reciente de todas, y fue lanzada en el año 2010, combina la tecnología bluetooth clásica con la conexión inalámbrica vía wifi, para dotar a los dispositivos en los que vienen instalados de una velocidad de emisión y transferencia de datos de nada más y nada menos que de 32Mb/s.. Esta nueva interfaz de bluetooth viene incluida en los más avanzados Smartphones y dispositivos tecnológicos de última generación.

BLUETOOTH V4.1

Los responsables del desarrollo de la especificación Bluetooth han presentado una nueva versión. La actualización 4.1 que incorpora novedades importantes de cara al usuario al facilitar la reconexión entre sus dispositivos una vez que estos salen y vuelven a entrar en el radio de acción, y que claramente orienta su uso al internet de las cosas.

Estas novedades facilitan la conexión y reconexión de dispositivos, como hemos dicho, sin la necesidad de que el usuario deba hacer nada, al menos una menor frecuencia de interacción. Además permitirá que un dispositivo funcione tanto como periférico y hub de datos a la par, se mejora el soporte para convivir con otros protocolos como LTE, intercambio de datos más eficientes, etc.

BLUETOOTH V4.2

De acuerdo con un comunicado de prensa oficial, la versión 4.2 de la especificación básica Bluetooth salió cerca del final de 2014. Director Ejecutivo del Bluetooth Special Interest Group (SIG) Mark Powell dice que la actualización 4.2 espera continuar haciendo Bluetooth Smart "de la mejor solución para

conectar toda la tecnología en su vida. "Bluetooth Core Especificación 4.2 esperanzas para dar a los desarrolladores y fabricantes más oportunidades de usar Bluetooth y construir una mejor experiencia de usuario para sus consumidores. (RODRIGUEZ, 2017)

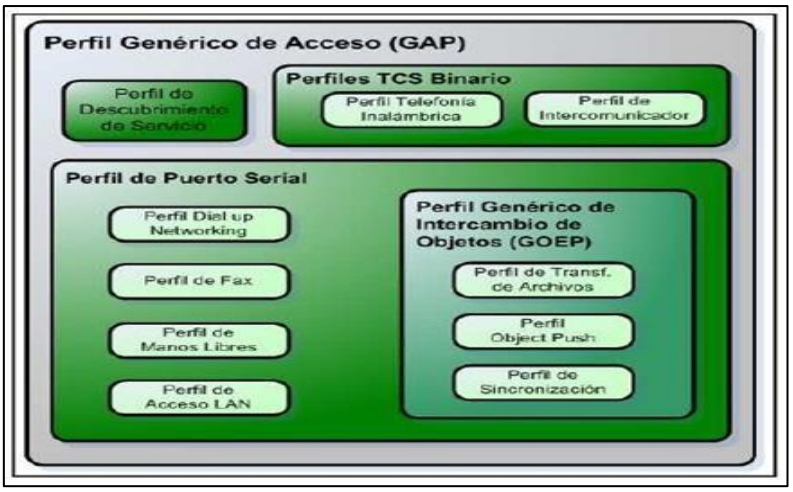
2.12.4. PERFILES DE BLUETOOTH

El estándar Bluetooth fue creado para ser usado por un gran número de fabricantes e implementado en áreas ilimitadas. Para asegurar que todos los dispositivos que usen Bluetooth sean compatibles entre sí son necesarios esquemas estándar de comunicación en las principales áreas. Para evitar diferentes interpretaciones del estándar Bluetooth acerca de cómo un tipo específico de aplicación debería ser implementado, el Bluetooth Special Interest Group (SIG), ha definido modelos de usuario y perfiles de protocolo. Un perfil define una selección de mensajes y procedimientos de las especificaciones Bluetooth y ofrece una descripción clara de la interfaz de aire para servicios específicos.

Un perfil puede ser descrito como una “rodaja” completa de la pila de protocolo. Existen cuatro perfiles generales definidos, en los cuales están basados directamente algunos de los modelos de usuario más importantes y sus perfiles. Estos cuatro modelos son: Perfil Genérico de Acceso (GAP), Perfil de Puerto Serial, Perfil de Aplicación de Descubrimiento de Servicio (SDAP) y Perfil Genérico de Intercambio de Objetos (GOEP).

A continuación se hace una breve descripción de estos y algunos otros perfiles Bluetooth. La Figura 2.48 muestra el esquema de los perfiles Bluetooth.

En ella se puede observar la jerarquía de los perfiles, como por ejemplo que todos los perfiles están contenidos en el Perfil Genérico de Acceso (GAP). (RODRIGUEZ, 2017)



48.FIGURA 2.48. PERFILES DE BLUETOOTH **(RODRIGUEZ, 2017)**

Perfil Genérico de Acceso (GAP) este perfil define los procedimientos generales para descubrir y establecer una conexión entre dispositivos Bluetooth. El GAP maneja el descubrimiento y establecimiento entre unidades que no están conectadas y asegura que cualquier par de unidades Bluetooth, cualquiera que sea su fabricante o aplicación, puedan intercambiar información a través de Bluetooth para descubrir qué tipo de aplicaciones soportan las unidades.

Perfil de Puerto Serial define los requerimientos para dispositivos Bluetooth, necesarios para establecer una conexión de cable serial emulada usando RFCOMM entre dos dispositivos similares. Este perfil solamente requiere soporte para paquetes de un slot, esto significa que pueden ser usadas tasas de datos de hasta 128 Kb/sl. RFCOMM es usado para transportar los datos de usuario, señales de control de modem y comandos de configuración. El perfil de puerto serial es dependiente del GAP.

Perfil de Aplicación de Descubrimiento de Servicio (SDAP) define los protocolos y procedimientos para una aplicación en un dispositivo Bluetooth donde se desea descubrir y recuperar información relacionada con servicios localizados en otros dispositivos. El SDAP es dependiente del GAP.

Perfil Genérico de Intercambio de Objetos (GOEP) define protocolos y procedimientos usados por aplicaciones para ofrecer características de intercambio de objetos. Los usos pueden ser, por ejemplo: sincronización, transferencia de archivos o modelo Object Push.

Los dispositivos más comunes que usan este modelo son: agendas electrónicas, PDAs, teléfonos celulares y teléfonos móviles. El GOEP es dependiente del perfil de puerto serial.

Perfil de Telefonía Inalámbrica este perfil define como un teléfono móvil puede ser usado para acceder a un servicio de telefonía de red fija a través de una estación base. Es usado para telefonía inalámbrica de hogares u oficinas pequeñas. El perfil incluye llamadas a través de una estación base, haciendo llamadas de intercomunicación directa entre dos terminales y accediendo adicionalmente a redes externas. Es usado por dispositivos que implementan el llamado "teléfono 3 en 1".

Perfil de Intercomunicador define los usos de teléfonos móviles, los cuales establecen enlaces de conversación directa entre dos dispositivos. El enlace directo es establecido usando señalización de telefonía sobre Bluetooth. Los teléfonos móviles que usan enlaces directos funcionan como walkie-talkies.

Perfil de Manos Libres este perfil define los requerimientos, para dispositivos Bluetooth, necesarios para soportar el uso de manos libres. En este caso el dispositivo puede ser usado como unidad de audio inalámbrico de entrada/salida. El perfil soporta comunicación segura y no segura.

Perfil Dial-up Networking define los protocolos y procedimientos que deben ser usados por dispositivos que implementen el uso del modelo llamado Puente Internet. Este perfil es aplicado cuando un teléfono celular o modem es usado como un modem inalámbrico.

Perfil de Fax este perfil define los protocolos y procedimientos que deben ser usados por dispositivos que implementen el uso de fax. En el perfil un teléfono celular puede ser usado como un fax inalámbrico.

Perfil de Acceso LAN este perfil define el acceso a una red de área local, LAN, usando el protocolo punto-a-punto, PPP, sobre RFCOMM. PPP es ampliamente usado para lograr acceder a redes soportando varios protocolos de red. El perfil soporta acceso LAN para un dispositivo Bluetooth sencillo, acceso LAN para varios dispositivos Bluetooth y PC-a-PC (usando interconexión PPP con emulación de cable serial).

Perfil Object Push define protocolos y procedimientos usados en el modelo object push. Este perfil usa el GOEP. En el modelo object push hay

procedimientos para introducir en el inbox, sacar e intercambiar objetos con otro dispositivo Bluetooth.

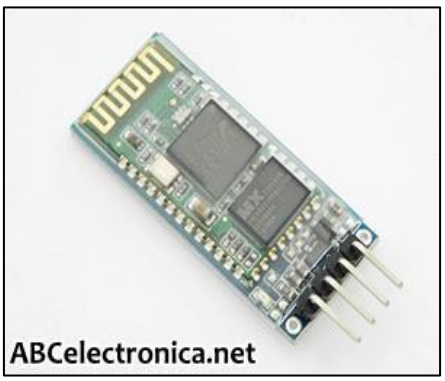
Perfil de Transferencia de Archivos este perfil define protocolos y procedimientos usados en el modelo de transferencia de archivos. El perfil usa el GOEP. En el modelo de transferencia de archivos hay procedimientos para chequear un grupo de objetos de otro dispositivo Bluetooth, transferir objetos entre dos dispositivos y manipular objetos de otro dispositivo. Los objetos podrían ser archivos o fólderes de un grupo de objetos tal como un sistema de archivos.

Perfil de Sincronización este perfil define protocolos y procedimientos usados en el modelo de sincronización. Éste usa el GOEP. El modelo soporta intercambios de información, por ejemplo para sincronizar calendarios de diferentes dispositivos. (RODRIGUEZ, 2017)

2.12.5. DISPOSITIVO BLUETOOTH HC06

El módulo a utilizar para la comunicación Bluetooth es el Modulo HC06 que a continuación se describirá.

En la Figura 2.49, 2.50 se muestra un módulo Bluetooth HC-06



49.FIGURA. 2.49. MÓDULO BLUETOOTH HC06 (RODRIGUEZ, 2017)

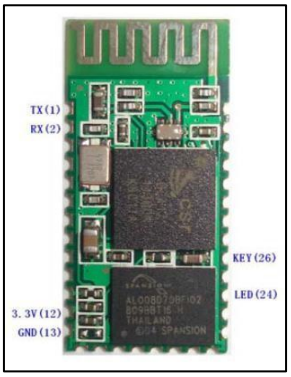
2.12.5.1. CARACTERÍSTICAS DEL MÓDULO HC06

- Modulo Bluetooth Slave HC-06
- Protocolo bluetooth: Bluetooth especificación V2.0+EDR
- Frecuencia: 2.4Ghz ISM Band
- Rango de baudios ajustable: 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200

- Default: Slave, 9600 baud rate, N, 8,1. Pincode 1234
- Distancia bluetooth: 10 metros
- Tamaño compacto

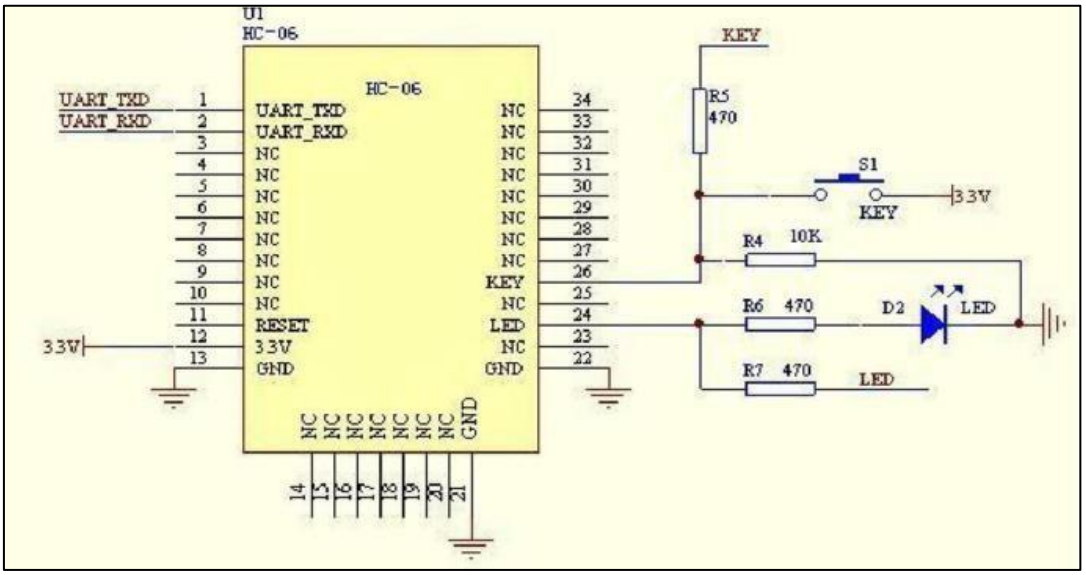
Una de las ventajas principales del **módulo HC-06**, además de su pequeño tamaño y sus buenas características de transmisión y recepción que le brindan un alcance muy amplio (por tratarse de un sistema local **Bluetooth**), es el **bajo consumo de corriente** que posee tanto en funcionamiento, como en modo de espera, es decir, alimentado con energía, pero sin conexión o enlace a otro dispositivo, por ejemplo, un móvil con SO **Android.**

Otra característica interesante de este módulo es que una vez que ha realizado un enlace con otro dispositivo es capaz de recordarlo en su memoria y no solicita validación alguna ("*1234*" por defecto), pero si se activa el pin 26 (*KEY*) hacia la tensión de alimentación, esta información se elimina y el **módulo HC-06** solicitará nuevamente la validación del enlace. Otro detalle particular es que su tensión de alimentación de 3,3Volts y su bajo consumo (8mA en transmisión/recepción activa) lo transforman en un dispositivo ideal para trabajar con microcontroladores de la misma tensión de alimentación, logrando de este modo equipos portátiles que pueden ser alimentados durante muchas horas por **baterías recargables o alcalinas AA**, demostrando características excepcionales en aplicaciones médicas, o para actividades recreativas donde la fuente energética debe ser liviana y portátil. (RODRIGUEZ, 2017)



50.FIGURA 2.50. MÓDULO BUETOOTH HC06 **(RODRIGUEZ, 2017)**

2.12.5.2. CONECTANDO EL MODULO BLUETOOTH HC-06 CON MICROCONTROLADOR



51.FIGURA 2.51. CONEXIONES DEL MÓDULO BUETOOTH HC06 **(RODRIGUEZ, 2017)**

El primer paso es reconocer que modulo tenemos, para esto debemos conectar la alimentación del módulo a 3.3V, después debemos buscar el dispositivo bluetooth ya sea con la PC o con un celular, el módulo HC-06 será encontrado con el nombre de **"linvor"**.

Ahora debemos configurar nuestro modulo, el HC-06 se puede configurar por medio de comandos **AT** y los valores que podemos modificar son el nombre del dispositivo, la contraseña (**PIN**) para realizar la conexión y el baudrate. Para que los comandos AT funcionen el modulo no debe estar apareado con el dispositivo maestro, debe ser configurado por medio de un microcontrolador o mediante un convertidor usb-serial y la terminal serie en una PC. (RODRIGUEZ, 2017)

Los comandos AT disponibles son los siguientes:

AT: Sirve como test de comunicación, responde con **OK**

AT+VERSION: Devuelve la versión del firmware del dispositivo, responde con **OKlinvorV1.5**

AT+NAME: Cambia el nombre del dispositivo, por ejemplo **AT+NAMEdispBT1** responde con **OKsetname** y ahora tendrá el nombre de**dispBT1**, el nombre es limitado a 20 caracteres.

AT+PINxxxx: Cambia el pin de seguridad de 4 digitos, podemos usar **AT+PIN0000** para setear el pin a 0000, responde con **OKsetPIN,** por default viene configurado 1234.

AT+BAUDx : Modifica el baudrate del dispositivo, x puede tomar los siguientes valores

1———1200
2———2400
3———4800
4———9600 (Default)
5———19200
6———38400
7———57600
8———115200
9———230400
A———460800
B———921600
C———1382400

Debes tomar en cuenta que el baudrate máximo que maneja una PC es de 115200, por lo que si estas configurando tu modulo por medio de esta y escoges un baudrate mayor a 115200 perderás la comunicación completamente con el dispositivo, si esto llega a suceder solo podrás reconfigurarlo por medio de un microcontrolador capaz de manejar tal velocidad mayor a 115200. Si la velocidad no es primordial en tu diseño maneja la velocidad de 9600 por default o en caso necesario la de 115200 como máximo. Para 9600 baudios usamos **AT+BAUD4** y responde **OK9600**

Una vez configurado el dispositivo lo podemos utilizar con un microcontrolador y realizar una comunicación serial de forma transparente.

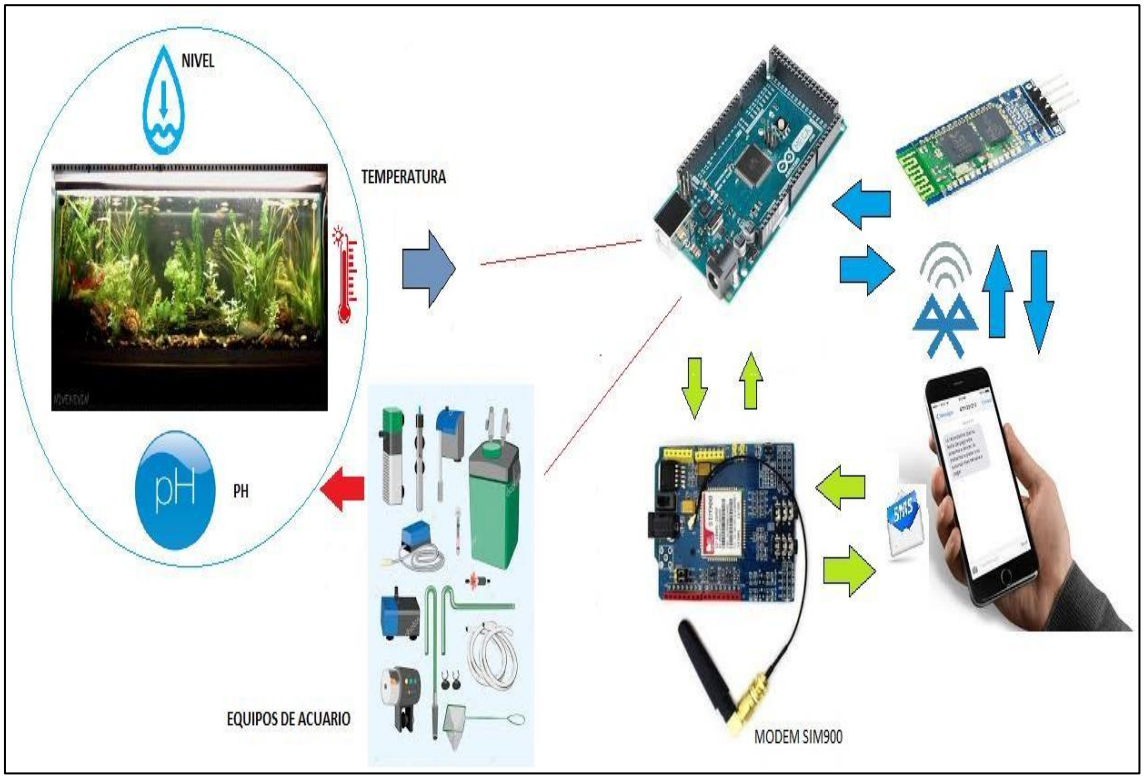
Nota: Si se está trabajando solo con el chip, debe tomar en cuenta que el fabricante toma el pin TX(1) como la entrada de datos que serán enviados posteriormente por Bluetooth y RX(2) donde salen los datos recibidos por Bluetooth. (RODRIGUEZ, 2017)

CAPITULO 3

DISEÑO DEL SISTEMA AUTOMATIZADO PARA CONTROL Y SUPERVISION DE UN ACUARIO CON TECNOLOGIAS INALAMBRICAS

3.1 DESCRIPCION GENERAL

En la Figura 3.1 se muestra el diagrama de bloques del sistema automatizado para el control y supervisión de un acuario con tecnologias inalambricas GPRS y BLUETOOTH, mediante el envio y recepción de comandos de texto.



52.FIGURA 3.1. DIAGRAMA DE BLOQUES DEL SISTEMA AUTOMATIZADO PARA CONTROL Y SUPERVISION DE UN ACUARIO CON TECNOLOGIAS INALAMBRICAS (PROPIO)

Se diseñará e implementará un módulo de control y supervisión basado en el ARDUINO MEGA 2560 que recogerá la información de las variables de nivel de agua, temperatura y PH, también se encargará de realizar ciertas tareas

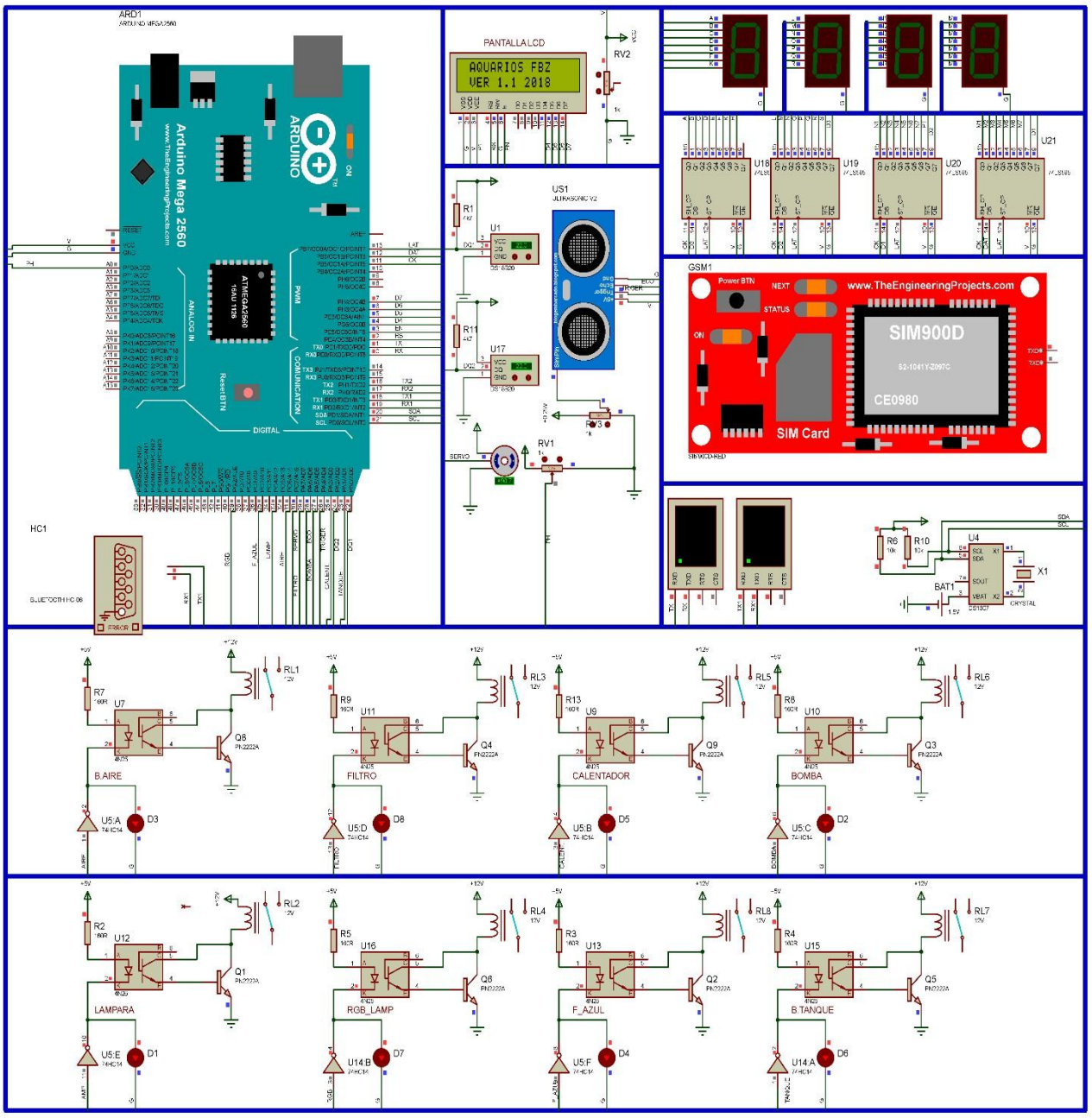
periódicas como encender y apagar equipos del acuario como las luces, filtros bombas de agua y compresores. El módulo de control de temperatura tomará el valor de sensores de temperatura para manejar al calentador del acuario, el módulo de supervisión del pH deberá permitir monitorizar el valor de pH en el acuario. Cuando las variables monitoreadas de temperatura, nivel y PH superen cierto rango deberá generar una señal de alarma enviando mensajes de texto a un número de celular específico. El sistema deberá permitir, por otro lado, supervisar y controlar el Nivel de agua en el acuario, y realizar los cambios de agua de manera automática al 25% o 50% del agua del acuario. Se implementara un dosificador de alimentos para peces basado en servo motor el cual se le podrá programar cuando y cuantas veces entregara comida en el acuario. Para la comunicación se utilizara módulos Bluetooth esclavo y modem GSM/GPRS SIM900, para lo cual se implementara los respectivos protocolos de comunicación con comandos tipo texto por ejemplo un comando seria 1234V, el cual solicita el estado de la variables monitoreadas.

3.2. COMPONENTES PRINCIPALES DEL SISTEMA

En la Figura 3.2 se muestra los componentes principales de la tarjeta electrónica para el sistema de monitoreo de amenazas fisicas en un centro de datos usando tecnologias inalambricas como GSM, para monitoreo usando de mensajes de texto.

Los componentes del sistema son:

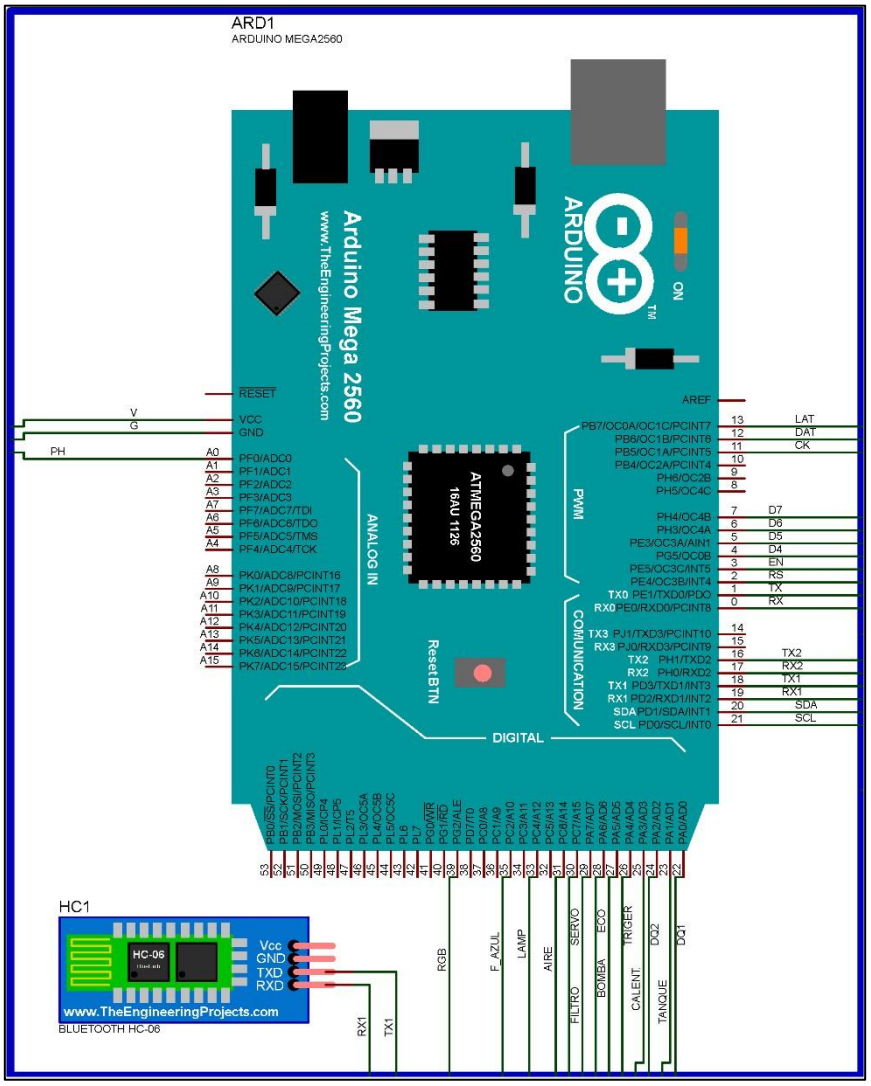
- ARDUINO MEGA 2560
- MODULO GSM/GPRS SIM900
- MODULO BLUETOOTH HC06
- SENSOR DE TEMPERATURA DS18B20
- SENSORE DE NIVEL HCSR04.
- SENSORES DE PH
- RELES DE ACTIVACION DE EQUIPOS DE ACUARIOS
- RELOJ TIEMPO REAL RTC DS1307
- DISPLAY CUATRO DIGITOS
- LCD I2C
- FUENTE DE ALIMENTACION



53.FIGURA 3.2 COMPONENTES PRINCIPALES DEL SISTEMA PROPUESTO. (PROPIO)

3.2.1. ARDUINO MEGA.

En la Figura 3.3 se muestran las conexiones para la tarjeta ARDUINO MEGA, el cual se encarga de leer el sensor digital de temperatura DS18B20, sensor de PH014 para ARDUINO, sensor de Nivel HCSR04, activar o desactivar relés que controlan los equipos de Acuario como son Bombas, Filtros.Visualizar los datos en display de 4 dígitos y LCD I2C enviar los datos vía puerto USB y también usando otros pines para comunicación serial con Modem GSM SIM900 para control y supervisión por mensajes de texto y Modulo Bluetooth HC06 para control y supervisión por mensajes por Bluetooth.



54.FIGURA 3.3 CONEXIONES DEL ARDUINO MEGA. (PROPIO)

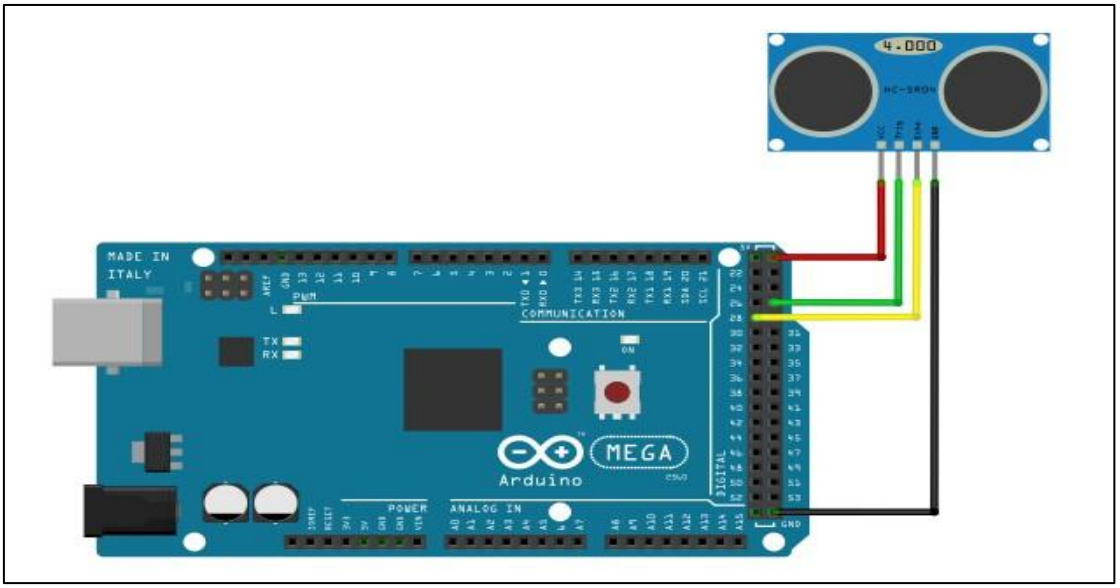
3.2.2. FUENTE DE ALIMENTACIÓN 12VDC - 5VDC

La fuente de alimentación está formada por regulador de voltaje al cual le ingresa 12voltios y a la salida se obtiene los 5 voltios para alimentar componentes de la tarjeta que necesitan 5 voltios. En la Figura 3.4 se visualiza la fuente de alimentación. La Tarjeta ARDUINO MEGA se alimenta directo con fuente de 12VDC 2A.

55.FIGURA 3.4 FUENTE DE ALIMENTACION. (PROPIO)

3.2.3. MODULO DE SENSOR DE NIVEL

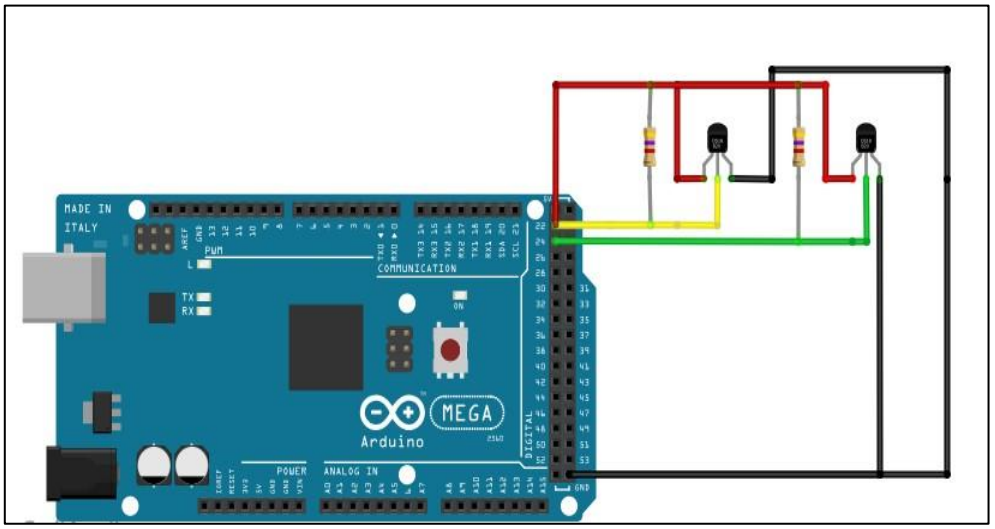
En la Figura 3.5 se muestra el circuito conexión del sensor de Nivel HCSR04, el cual es ubicado 25 cm por encima del acuario y al centro teniendo en cuenta su ángulo de trabajo de 15°.



56.FIGURA 3.5 CONEXIÓN DEL SENSOR DE NIVEL HCSR04. (PROPIO)

3.2.4. SENSORES DE TEMPERATURA

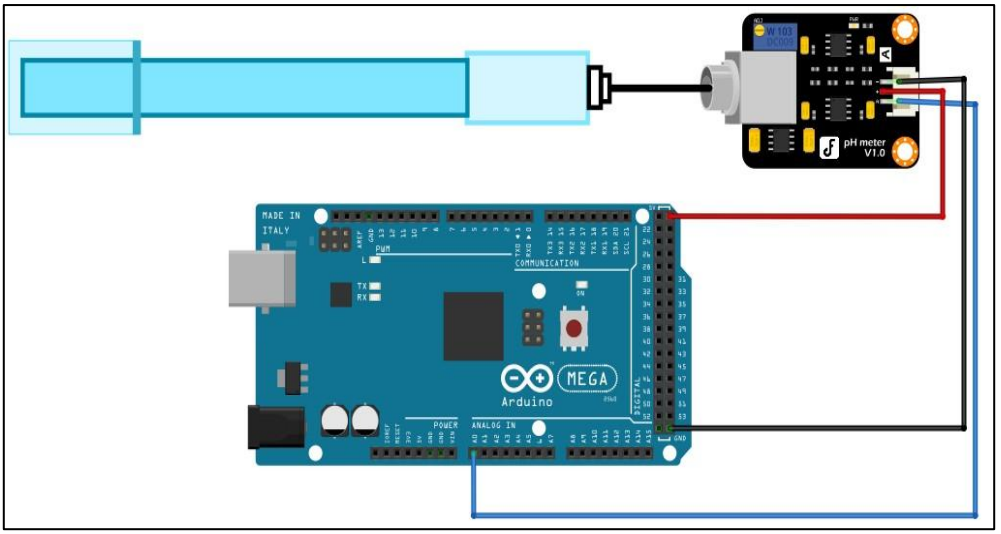
En la Figura 3.6 se muestra como está conectado los sensores de temperatura DS18B20 con el ARDUINO MEGA. Un sensor será ubicado en el interior del acuario y otro en el exterior.



57.FIGURA 3.6. CONEXIÓN DE SENSORES DS18B20 (PROPIO)

3.2.5. SENSOR DE PH

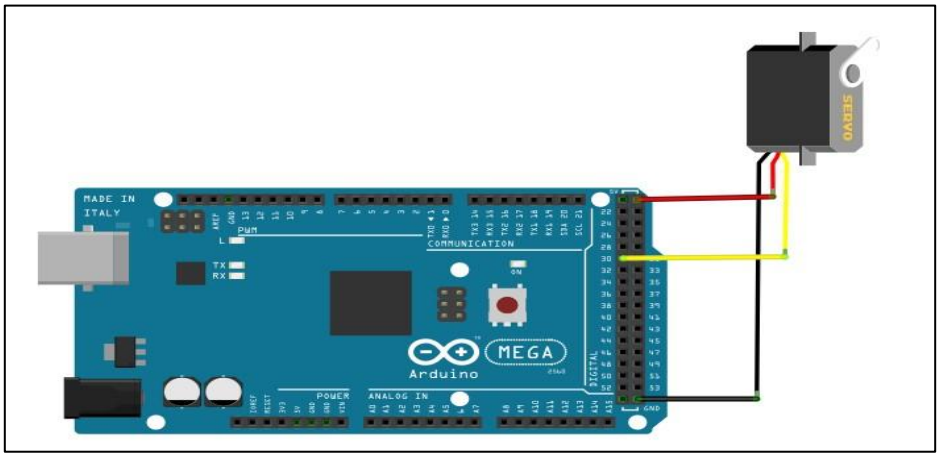
En la Figura 3.7. Se muestra como están conectado el sensor de PH con el ARDUINO MEGA. Para leer el PH se usara la entrada analógica A0.



58.FIGURA 3.7. CONEXIÓN DEL SENSOR DE PH PARA ARDUINO (PROPIO)

3.2.6. CONTROL DE SERVO MOTOR PARA ALIMENTADOR

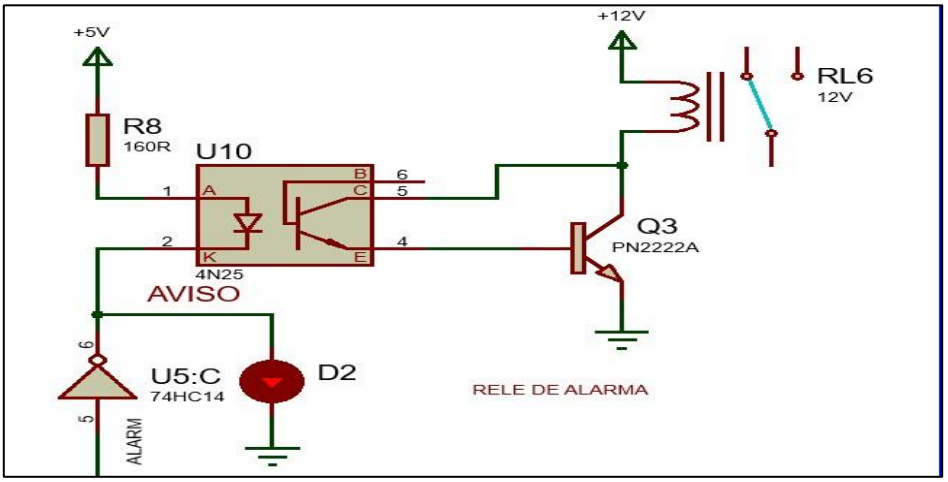
En la Figura 3.8 se muestra la conexión necesaria para el servomotor que se encarga del alimentador de los peces.



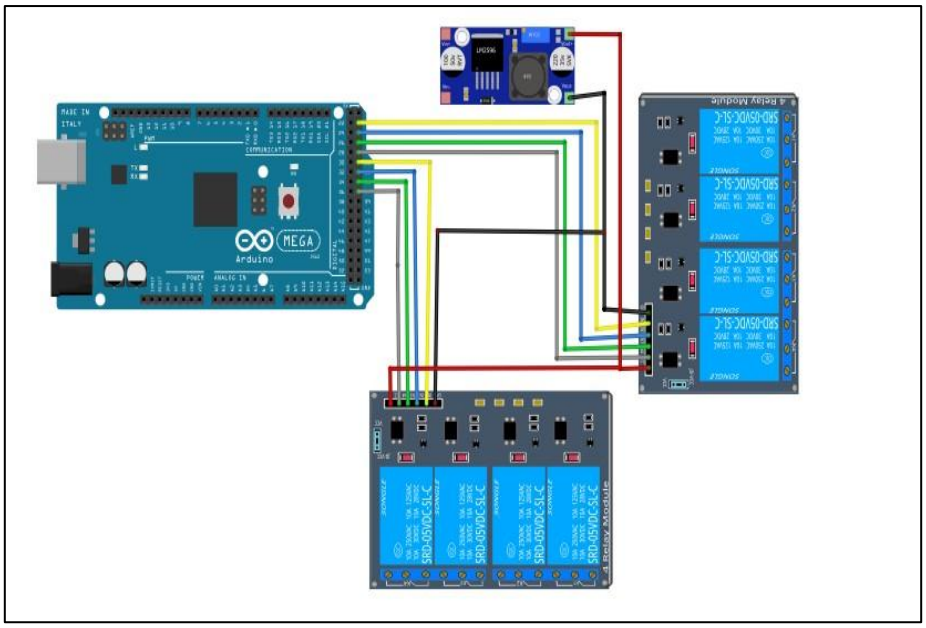
59.FIGURA 3.8. Conexión del Servomotor. (PROPIO)

3.2.7. CIRCUITO PARA ACTIVACION DE EQUIPOS DEL ACUARIO

En la Figura 3.9 y 3.10 se muestra el circuito para la activación mediante 8 relés de 12V o 5V para Bobina y contactos de 220VAC 10A para control ON/OFF de equipos de ACUARIO que son: Bomba de Tanque, Bomba de agua de acuario, bomba de aire, Filtros, lámpara luz blanca, lámpara luz azul, lámpara RGB, Termostato.



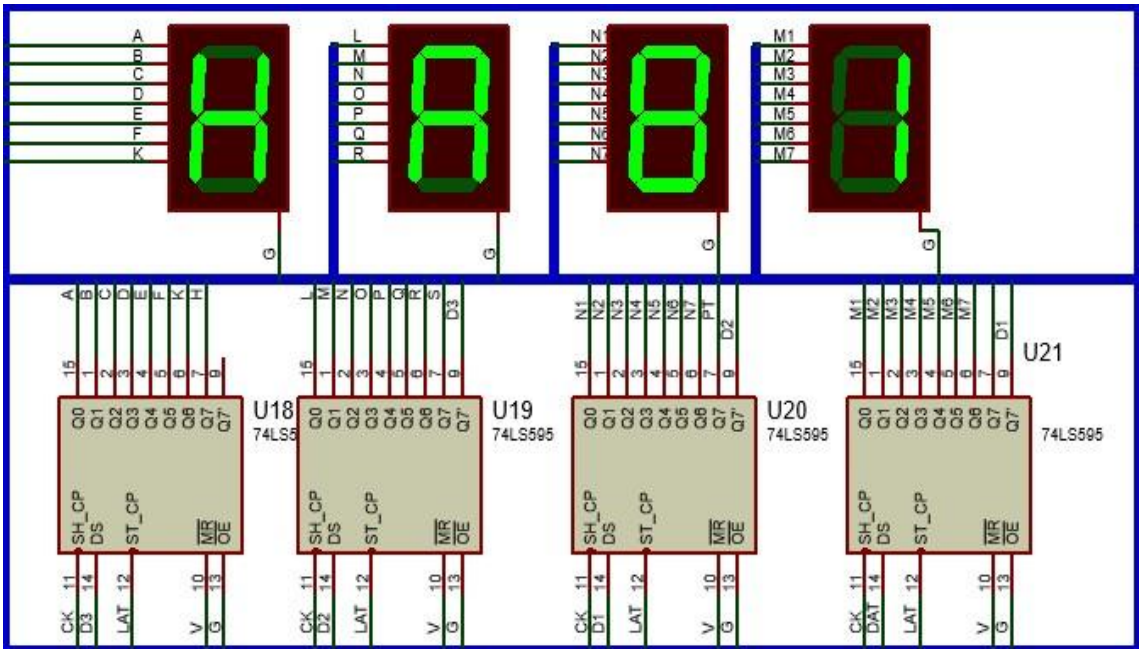
60. FIGURA 3.9 CIRCUITO DE ACTIVACION POR RELE DE EQUIPOS DE ACUARIO (PROPIO)



61. FIGURA 3.10 CIRCUITO DE ACTIVACION DE 8 EQUIPOS PARA ACUARIO (PROPIO)

3.2.8. CIRCUITO DE VISUALIZACION EN DISPLAY DE 4 DIGITOS

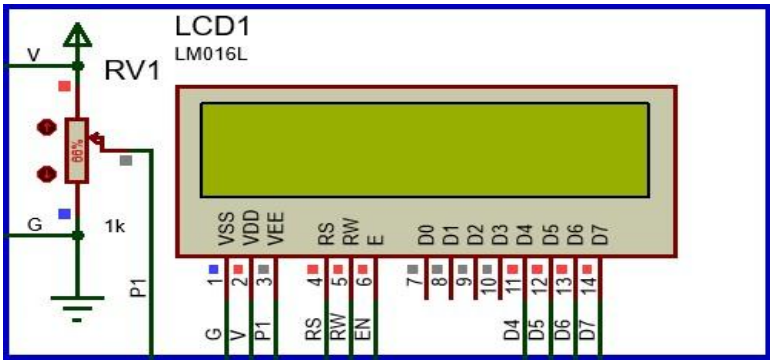
En la Figura 3.11 se muestra el circuito basado en el registro 74HC595 para la visualización del humedad y la temperatura.



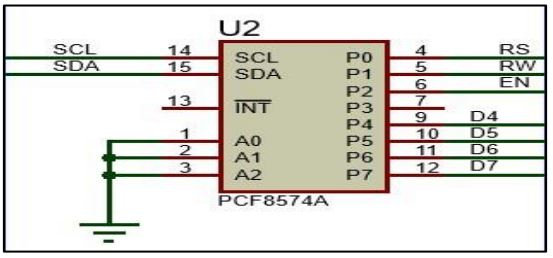
62. FIGURA 3.11 CIRCUITO DE VISUALIZACION DE DATOS EN DISPLAY DE 4 DIGITOS (PROPIO)

3.2.9. CONEXIÓN DEL LCD 16X2

En la Figura 3.12 se muestra las conexiones del LCD16X2, este es conectado con el C.I. PCF8574A (Figura 3.13) para poder tener comunicación I2C con ARDUINO.



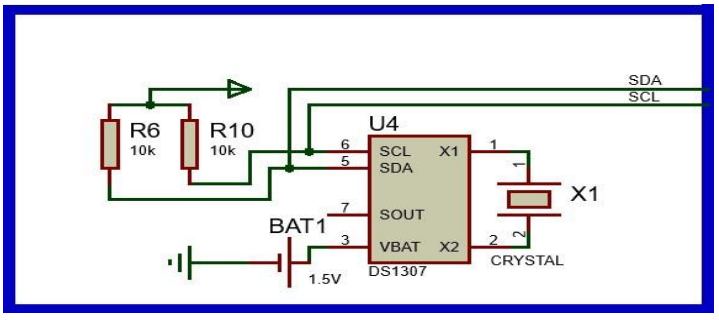
63. FIGURA 3.12 CONEXIÓN DEL LCD (PROPIO)



64. FIGURA 3.13 CONEXIÓN I2C PARA LCD 16X2. (PROPIO)

3.2.10. MODULO RTC DS1307 (RELOJ TIEMPO REAL)

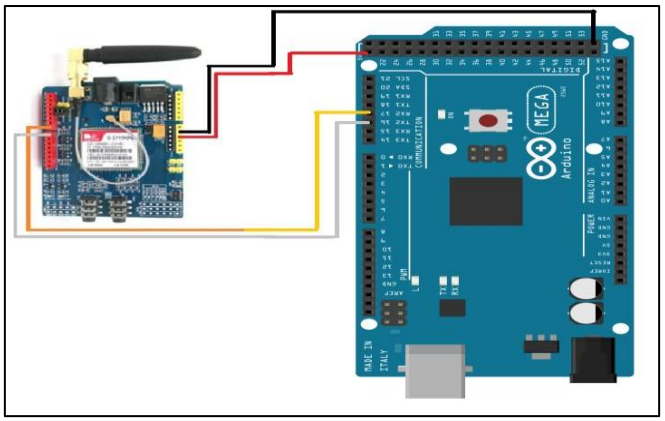
En la Figura 3.14 se muestra como están conectado el Modulo GSM/GPRS SIM900 con el ARDUINO. Este Modulo se encarga de recibir el comando de mensaje de texto para enviar también mediante un mensaje de texto los estados de los sensores.



65.FIGURA 3.14 CONEXIÓN DEL RTC DS1307. (PROPIO)

3.2.10. MODULO GSM/GPRS

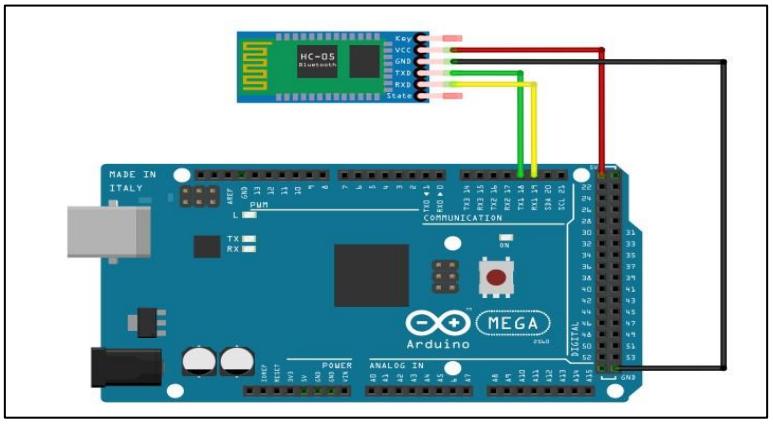
En la Figura 3.15 se muestra como están conectado el Modulo GSM/GPRS SIM900 con el ARDUINO MEGA. Este Modulo se encarga de recibir el comando de mensaje de texto para enviar también mediante un mensaje de texto los estados de los sensores o activar los equipos del acuario. Los comandos se visualizan más adelante en una tabla.



66 FIGURA 3.15. CONEXIÓN DEL MODULO GSM/GPRS SIM900. (PROPIO)

3.2.11. MODULO BLUETOOTH

En la Figura 3.16 se muestra como están conectado el Modulo BLUETOOTH HC06 con el ARDUINO MEGA. Este Modulo se encarga de recibir el comando formato de texto para enviar también mediante un mensaje de texto los estados de los sensores o activar equipos del acuario. Los comandos e visualizan más adelante en una tabla.



67.FIGURA 3.16. CONEXIÓN DEL MODULO BLUETOOTH HC06. (PROPIO)

3.3. PROTOCOLOS DE COMUNICACIÓN

Para la comunicación entre la tarjeta electrónica ARDUINO MEGA 2560 y el modem SIM900 así como la comunicación por BLUETOOTH se estableció el mismo protocolo basado en comando ASCII que se describen en la Tabla3.1 y Tabla 3.2.

ITEM	COMANDO	DESCRIPCION
1	1234I	Cambio de agua al 25%
2	1234J	Cambio de agua al 50%
3	1234A	Enciende bomba de tanque de almacenamiento
4	1234a	Apaga bomba de tanque de almacenamiento
5	1234B	Enciende calentador
6	1234b	Apaga Calentador
7	1234O	Enciende bomba de agua de acuario
8	1234o	Apaga bomba de agua de acuario
9	1234D	Enciende filtros
10	1234d	Apaga filtros
11	1234E	Enciende bomba de aire
12	1234e	Apaga bomba de aire
13	1234R	Enciende lampara de luz blanca
14	1234r	Apaga lampara de luz blanca
15	1234G	Enciende luz de noche
16	1234g	Apaga luz de noche
17	1234H	Enciende lampara RGB
18	1234h	Apaga lampara RGB
19	1234L	Apaga todos los equipos
20	1234SN	Entrega comida donde "N" es el numero de porciones
21	1234x	Armar alarma de temperatura y nivel
22	1234X	Desarmar alarma de temperatura y nivel
23	1234y	Armar alarma de PH
24	1234Y	Desarmar alarma de PH
25	12348	Verificar estado de equipos de acuario
26	1234z	Verificar la Hora del RTC del equipo
27	1234p	Verificar horas programadas de comidas
28	1234Q	Verificar valores seteados de las variables en acuario
29	1234$	Verifica numero de celular programado para recepcion de alarmas
30	1234V	Envia valores de las variable de temperatura , nivel y PH
31	0801M	Programa primera comida a las 08:01
32	1700N	Programa segunda comida a las 17:00
33	0607K	Poner a la hora el RTC 06:07
34	951551591X	Se programa al celular 951551591 para recepcion de alarmas
35	23TX	Temperatura de referencia 23°
36	36NX	Nivel de referencia 36 cm
37	7.5P	PH de referencia 7.5

TABLA N° 3.1. COMANDOS GSM (PROPIO)

ITEM	COMANDO	DESCRIPCION
1	1234A	Cambio de agua al 25%
2	1234B	Cambio de agua al 50%
3	A	Enciende bomba de tanque de almacenamiento
4	a	Apaga bomba de tanque de almacenamiento
5	B	Enciende calentador
6	b	Apaga Calentador
7	O	Enciende bomba de agua de acuario
8	o	Apaga bomba de agua de acuario
9	D	Enciende filtros
10	d	Apaga filtros
11	E	Enciende bomba de aire
12	e	Apaga bomba de aire
13	F	Enciende lampara de luz blanca
14	f	Apaga lampara de luz blanca
15	G	Enciende luz de noche
16	g	Apaga luz de noche
17	H	Enciende lampara RGB
18	h	Apaga lampara RGB
19	L	Apaga todos los equipos
20	SN	Entrega comida donde "N" es el numero de porciones
21	x	Armar alarma de temperatura y nivel
22	X	Desarmar alarma de temperatura y nivel
23	y	Armar alarma de PH
24	Y	Desarmar alarma de PH
25	1234E	Verificar estado de equipos de acuario
26	z	Verificar la Hora del RTC del equipo
27	p	Verificar horas programadas de comidas
28	Q	Verificar valores seteados de las variables en acuario
29	$	Verifica numero de celular programado para recepcion de alarmas
30	1234G	Envia valores de las variable de temperatura , nivel y PH
31	0801U	Programa primera comida a las 08:01
32	1700Y	Programa segunda comida a las 17:00
33	0607X	Poner a la hora el RTC 06:07
34	951551591X	Se programa al celular 951551591 para recepcion de alarmas
35	23TX	Temperatura de referencia 23°
36	36NX	Nivel de referencia 36 cm
37	7.5P	PH de referencia 7.5

TABLA N° 3.2. COMANDOS BLUETOOTH (PROPIO)

3.4. CODIGO DEL PROGRAMA PARA AURDUINO UNO

A continuación se presentara partes del código del programa de la tarjeta ARDUINO MEGA 2560, el código completo se encuentra a su disposición en el CD de la presente tesis.

```
//*******************LIBRERIAS HA UTILIZAR*********************************************************************
#include <OneWire.h>                //Incluir libreria para sensor DS18B20
#include <DallasTemperature.h>      //Incluir libreria para sensor DS18B20
#include <EEPROM.h>                 //Incluir libreria para memoria EEPROM
#include <Wire.h>
#include "RTClib.h"                 //Incluir libreria para RTCS DS1307
#include <Servo.h>                  //Incluir libreria del SERVO
#include <avr/wdt.h>                //Incluir la librería de ATmel para Whach Dog Timer para RESET
#include <SD.h>                     //Incluir libreria para leer y escribir memoria SD
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x3F,16,2);
//*****************************************************************************************************************
//DEFINICION DE NOMBRES DE LOS 8 RELES QUE MANEJAS LOS EQUIPOS DEL ACUARIO
#define bomba_tanque 23  //Relel
#define termostato 25 //Rele2
#define bomba 27//Rele3
#define filtrol 29//Rele4
#define bomba_airel 31//Rele5
#define lampara 33//Rele6
#define foquito_azul 35//Rele7
#define rgblampara 39//Rele8
#define boya 4//boya

//*****************************************************************************************************************
```

68 FIGURA 3.17. PARTE DEL CODIGO DE CONFIGURACION Y ASIGNACION DE VARIABLES (PROPIO)

```
//*****************************************************************************************************************
// Configuracion de pines para el LCD
//LiquidCrystal lcd(2, 3, 4, 5, 6, 7); //    ( RS, EN, d4, d5, d6, d7)
// NOMBRE DEL SERVO: servoMotor
Servo servoMotor;
//*****************************************************************************************************************
//DECLARACIONES PARA GRABACION DE ARCHIVO EN MEMORIA SD
File logFile;
File dataFile;
String dataLine;
int cc=0;
//*****************************************************************************************************************
//DECLARACIONES PARA RTC DS1307
RTC_DS1307 rtc;
int segundo,minuto,hora,dia,mes;
long anio; //variable año
DateTime HoraFecha;
int reloj=1;
//*****************************************************************************************************************
//DECLARACIONES PARA SENSORES DE TEMPERATURA DS18B20B
OneWire ourWire1(22);                //Se establece el pin 2  como bus OneWire
OneWire ourWire2(24);                //Se establece el pin 3  como bus OneWire
DallasTemperature sensors1(&ourWire1); //Se declara una variable u objeto para nuestro sensor1
DallasTemperature sensors2(&ourWire2); //Se declara una variable u objeto para nuestro sensor2
float temp1,temp2;
```

69 FIGURA 3.18. PARTE DEL CODIGO DE CONFIGURACION Y ASIGNACION DE VARIABLES (PROPIO)

```
//*************************************************************************************************************
//DECLARACIONES PARA SENSOR DE PH
const int analogInPin = A0;
int sensorValue = 0;
unsigned long int avgValue;
float b;
int buf[10],temp;
float pHVol;
float phValue;
//*************************************************************************************************************
//DECLARACION DEVARIABLES PARA USAR CON PUERTOS SerialES PARA BLUETOOTH,GSM Y USB-SERIAL ARDUINO
int trama=0;
String p="";
char inicio_clave;
char clave1;
String clave = "";
char num;
char letra;
char letras[12];
int c;
char incoming_char = 0; //Variable que guarda los caracteres que envia el Serial2
char incoming_char1 = 0; //Variable que guarda los caracteres que envia el SIM900
char incoming_char2 = 0;
String numero = "";
```

70 .FIGURA 3.19. PARTE DEL CODIGO DE CONFIGURACION Y ASIGNACION DE VARIABLES (PROPIO)

```
//*********************************RUTINAS DE CONFIGURACIONES INICIALES*****************************
void setup() {
  wdt_disable(); // Desactiva la función mientras se configura el tiempo en el que se reseteara
  wdt_enable(WDTO_8S); // Configuramos el contador de tiempo para que se reinicie en 2s
  delay(100);
  Serial.begin(9600); // Velocidad del puerto serial/USB del arduino
  Serial1.begin(9600);// Velocidad para el puerto que utilizara el modulo bluetooth
  Serial2.begin(19200);//Velocidad para conexión con el SIM900
  servoMotor.attach(30);// Servomotor conectado al pin 30
  // Inicializar el LCD
  lcd.init();
  lcd.clear();
  lcd.backlight(); //Encender la luz de fondo.
  lcd.setCursor(0,0);
  lcd.print(" AQUARIOS FBZ");
  lcd.setCursor(0,1);
  lcd.print(" VER 1.1 2018") ;
  delay(1000);
  //*************************************************************************************
  sensors1.begin();   //Se inicia el sensor 1 de temperatura interior acuario
  sensors2.begin();   //Se inicia el sensor 2 de temperatura exterios de acuario
  //*************************************************************************************
  rtc.begin(); //Inicializamos el RTC
  //*************************************************************************************
  // Se configura reles como salidas digitales
  pinMode(bomba_tanque, OUTPUT);
  pinMode(termostato, OUTPUT);
```

71 FIGURA 3.20. PARTE DEL CODIGO DE CONFIGURACIONS INICIALES. (PROPIO)

```
Serial.print("SET POINT TEMPERATURA ACUARIO:");
Serial.println(ts);
Serial.print("TEMP_min:");
Serial.println(tsmin);
Serial.print("TEMP_max:");
Serial.println(tsmax);
Serial.print("SET POINT NIVEL DE ACUARIO:");
Serial.println(ns);
Serial.print("NIVEL_min:");
Serial.println(nsmin);
Serial.print("NIVEL_max:");
Serial.println(nsmax);
Serial.print("NIVEL 25%:");
Serial.println(ns25);
Serial.print("NIVEL 50%:");
Serial.println(ns50);
Serial.print("NIVEL 75%:");
Serial.println(ns75);
Serial.print("SET POINT PH ACUARIO:");
Serial.println(ps);
Serial.print("PHmin:");
```

72 FIGURA 3.21. PARTE DEL CODIGO DONDE SE ENVIA LOS VALORES SETEADOS DE VARIABLES. (PROPIO)

```
void loop() {
  sms();
  serial_bluetooth();
  serial_usb();
  if (sm==0){
    tiempo();
    temperatura();
    nivel();
    ph();
    digitos();
    ver_lcd();
    delay(100);
    control_nivel();
    control_temperatura();
    niveles_agua();
   // enviar_blue();
   if (captura==1){
      estado_equipos();
      enviar_serial();
   }
   if (grabar_sd==1){
      estado_equipos();
      grabar_memoria();
     }
    alarmas();
    phalarmas();
    leer_boya();
  }
```

73. FIGURA 3.22. PROGRAMA PRINCIPAL QUE SE EJECUTA CICLICAMENTE (PROPIO)

```
//**********************LEER TEMPERATURA SENSORES DS18B20*****************************
void temperatura(){
  wdt_reset(); // Actualizar el watchdog para que no produzca un reinicio
  leer_temp:
  sensors2.requestTemperatures();   //Se envía el comando para leer la temperatura
  temp1= sensors2.getTempCByIndex(0); //Se obtiene la temperatura en °C del sensor 1
 // Serial.println(temp1);
  delay(500);
  if((temp1== -127) or(temp1== 85)){
   // Serial.println("error de sensor de temperatura de acuario");
    digitalWrite(termostato, HIGH);
    digitalWrite(filtrol, LOW);
    sms();
    tiempo();
    leer_boya();
    serial_bluetooth();
    goto leer_temp;
  }
  sensors1.requestTemperatures();   //Se envía el comando para leer la temperatura
  temp2= sensors1.getTempCByIndex(0); //Se obtiene la temperatura en °C del sensor 2
}
//*******************************************************************************************
```

74. FIGURA 3.23. CODIGO PARA LEER SENSORES DE TEMPERATURA DS18B20. (PROPIO)

```
void nivel(){
  wdt_reset(); // Actualizar el watchdog para que no produzca un reinicio
  for(int i=0;i<10;i++)
    {
      srhc04();
      buf1[i]=distance;
      delay(10);
    }
  for(int i=0;i<9;i++)
    {
      for(int j=i+1;j<10;j++)
        {
          if(buf1[i]>buf1[j])
          {
            temp5=buf1[i];
            buf1[i]=buf1[j];
            buf1[j]=temp5;
          }
        }
    }
  avgValue1=0;
  for(int i=2;i<8;i++){
    avgValue1+=buf1[i];
  }
  nivel_ok=avgValue1/6;
  delay(20);
}
```

75. FIGURA 3.24 CODIGO PARA LEER SENSOR DE NIVEL HCSR04. (PROPIO)

```
void ph(){
  wdt_reset(); // Actualizar el watchdog para que no produzca un reinicio
  for(int i=0;i<10;i++)
    {
      buf[i]=analogRead(analogInPin);
      delay(10);
    }
  for(int i=0;i<9;i++)
    {
      for(int j=i+1;j<10;j++)
        {
          if(buf[i]>buf[j])
          {
            temp=buf[i];
            buf[i]=buf[j];
            buf[j]=temp;
          }
        }
    }
  avgValue=0;
  for(int i=2;i<8;i++){
    avgValue+=buf[i];
  }
    pHVol=(float)avgValue*5.0/1024/6;
    phValue = -5.70 * pHVol + 21.34;
    delay(20);
}
```

76 FIGURA 3.25. PARTE DEL CODIGO PARA LEER SENSOR DE PH. (PROPIO)

```
//*****************************LCD 16X2************************************
// VISUALIZAR EN EL LCD DE 16X2 LOS VALORES DE LAS VARIABLES DEL ACUARIO
void ver_lcd(){
  wdt_reset(); // Actualizar el watchdog para que no produzca un reinicio
  if (reloj==1){
      lcd.init();
      lcd.clear();
      lcd.setCursor(0,0);
      lcd.print("A:");
      lcd.setCursor(2, 0);     // Ponte en la line 1, posicion
      lcd.print(temp1) ;
      lcd.setCursor(8,0);
      lcd.print("E:");
      lcd.setCursor(10, 0);     // Ponte en la line 1, posicion 1
      lcd.print(temp2) ;

      lcd.setCursor(0,1);   //Set Cursor again to first column of second row
      lcd.print(nivel_ok); //Print measured distance
      lcd.print("cm");  //Print your units.
      lcd.setCursor(8,1);
      lcd.print("PH:");
      lcd.setCursor(11,1);   //Set Cursor again to first column of second row
      lcd.print(phValue); //Print measured distance
      delay(500);
  }
```

77 FIGURA 3.26. VER LOS VALORES DE SENSORES EN EL LCD. (PROPIO)

```
//*************************************************************************************************
void ver_display(){
 //OBTENER LOS DIGITOS UNIDAD , CENTENA Y DECENA
  C = I / 100;  // 'I=123 C=123/100=1
  D = C * 100;  //'D=100
  D = I - D;    //'D=123-100=23
  De = D;
  D = D / 10;   //'D=23 D=23/10=2
  U = D * 10;   //'U=2*10=20
  U = De - U;   //'U=23-20=3
  //PRIMERO ENVIA LA PRIMERA LETRA SEGUN VARIABLE
  shiftOut(dataPin, clockPin, MSBFIRST, N1);
  C=C+100; // LOS VALORES DEL DISPLAY ESTAN GRABADOS APARTIR DE LA DIRECCION 100
  D=D+100;
  U=U+100;
  //SEGUNDO ENVIA LA SEGUNDA LETRA SEGUN VARIABLE
  shiftOut(dataPin, clockPin, MSBFIRST, N2);
  //COMO VARIABLE A MEDIR NO SUPERA LOS DIGITOS  SE ENVIA LA DECENA
  // Y LUEGO LA UNIDAD
  N=EEPROM.read(D);
  shiftOut(dataPin, clockPin, MSBFIRST, N);
  N=EEPROM.read(U);
  shiftOut(dataPin, clockPin, MSBFIRST, N);
  digitalWrite(latchPin, LOW);
  digitalWrite(latchPin, HIGH);
}
//*************************************************************************************************
```

78. FIGURA 3.27. VER HUMEDAD O TEMPERATURA EN DISPLAY DE 4 DIGITOS. (PROPIO)

```
//*******************RUTINA DE RX/TX POR MENSAJES DE TEXTO*********************************
void sms(){
  if (Serial2.available() >=2)
    {
      sm=1;//deshabilita rutinas de programa principal
      c=0;
      incoming_char = Serial2.read(); //Get the character from the cellular Serial2 port.
      if(incoming_char == 'K'){
        Serial2.end();
        delay(1000);
        Serial2.begin(19200);
        sm=0;
      }
      if((trama == 0) && (incoming_char == 'C')){
        trama = 1;
        p="";
        clave="";
       }
      if((trama == 1) && (incoming_char == 'M')){
        trama = 2;
      }
      if((trama == 2) && (incoming_char == 'T')){
        trama = 3;
      }
```

79. FIGURA 3.28. CODIGO PARA CONTROL Y SUPERVISION POR SMS USANDO EL MODEM SIM900. (PROPIO)

```
        numero2=p.substring(3);
       }
      // Serial.print("Numero telefonico: ");
      // Serial.println(p);  // imprimo en el puerto serial el número telefónico.
      // Serial.println(numero2);
     }
   // El mensaje es capturado a partir del caracter salto de línea \n.
     if ((trama==6)&&(incoming_char=='\n')){
       while(Serial2.available()>0){
         clavel = Serial2.read();
         clave += clavel;
         trama = 0;
        // Serial.println(clave);
         sm=0; // ejecuta todas la rutinas en programa principal
       }
       //Serial.println(clave);  // Imprimo el mensaje enviado desde un remitente
       clave.toCharArray(letras, 12);
     }
     delay(10);
     //sm=0;
  }
/******COMANDOS PARA ON/OFF EQUIPOS FORMATO 1234A  1234B 1234a....***********
 if ((letras[0]=='1')&&(c==0)){
   if ((letras[4]=='I')&&(c==0)){ //cambio de agua al 25% COMANDO: 1234I
         fase=3; //cambio de agua al 25%
```

80. FIGURA 3.29. CODIGO PARA LEER PUERTO DE COMUNICACIÓN DEL MODEM SIM900. (PROPIO)

```
          }
      if ((letras[4]=='J')&&(c==0)){ //cambio de agua al 50% COMANDO: 1234J

          fase=5; //cambio de agua al 50%
          cambioagua=1;
          digitalWrite(termostato, HIGH);
          inicializa();
        }
    if ((letras[4]=='A')&&(c==0)){ // ON BOMBA DE TANQUE COMANDO: 1234A
          digitalWrite(bomba_tanque, LOW);
          inicializa();
        }
    if ((letras[4]=='a')&&(c==0)){ // OFF BOMBA DE TANQUE COMANDO: 1234a
          digitalWrite(bomba_tanque, HIGH);
          inicializa();
        }
    if ((letras[4]=='B')&&(c==0)){ // ENCIENDE TERMOSTATO COMANDO: 1234B
          digitalWrite(termostato, LOW);
          inicializa();
        }
    if ((letras[4]=='b')&&(c==0)){ // APAGA TERMOSTATO    COMANDO: 1234b
          digitalWrite(termostato, HIGH);
          inicializa();
        }
```

81. FIGURA 3.30. CODIGO PARA LEER COMANDOS RECIBIDOS POS SMS. (PROPIO)

```
//********************* ENVIA POR SMS LAS VARAIBLES CON SU SET POINT*********************
if ((letras[4]=='Q')&&(c==0)){// ENVIA VALORES SETEADOS DE VARIABLES COMANDO: 1234Q
      Serial2.print("AT+CMGS=");
      Serial2.print((char)34);
      Serial2.print(numero2);
      Serial2.println((char)34);
      delay(1000);
      Serial2.print("SET TEMP:");
      Serial2.println(ts);
      Serial2.print("T_min:");
      Serial2.println(tsmin);
      Serial2.print("T_max:");
      Serial2.println(tsmax);
      Serial2.print("SET NIVEL:");
      Serial2.println(ns);
      Serial2.print("N_min:");
      Serial2.println(nsmin);
      Serial2.print("N_max:");
      Serial2.println(nsmax);
      Serial2.print("N_25%:");
      Serial2.println(ns25);
      Serial2.print("N_50%:");
      Serial2.println(ns50);
      Serial2.print("N_75%:");
      Serial2.println(ns75);
      Serial2.print("SET PH:");
      Serial2.println(ps);
      Serial2.print("PHmin:");
      Serial2.println(psmin);
```

82. FIGURA 3.31. CODIGO QUE ENVIA VALORES SETEADOS POR SMS (COMANDO SMS: 1234Q). (PROPIO)

```
//************ENVIA NUMERO DE CELULAR PROGRAMADP QUE RECIBIRA ALARMAS**********
 if (letras[4]=='$'){//
  numero="";
  n=89;
  for (k=80; k<n; k++) // se lee caracter (valor asccii) por caracter guardado en memoria eem
    {
      incoming_char =EEPROM.read(k);// se lee memoria y se guarda en la variable valorl
      numero = numero + incoming_char;
      delay(50);
    }
      Serial2.print("AT+CMGS=");
      Serial2.print((char)34);
      Serial2.print(numero2);
      Serial2.println((char)34);
      delay(1000);
      Serial2.print("CELULAR:");
      Serial2.println(numero);
      delay(100);
      Serial2.println((char)26); //Comando de finalización ^Z
      delay(3000);  // Esperamos un tiempo para que envíe el SMS
      inicializa();
 }
```

83. FIGURA 3.32. CODIGO QUE ENVIA NUMERO DE CELULAR PROGRAMADO POR SMS (COMANDO SMS: 1234$) (PROPIO)

```
//*********************COMANDO PARA SOLICITAR VALOR DE VARIABLES POS SMS*************
   if ((letras[4]=='V')&&(c==0)){// ENVIA VALORES DE LOS SENSORES COMANDO: 1234V
      Serial2.print("AT+CMGS=");
      Serial2.print((char)34);
      Serial2.print(numero2);
      Serial2.println((char)34);
      delay(100);
      Serial2.print("NIVEL ACUARIO:");  // Texto del SMS
      Serial2.println(nivel_ok);
      Serial2.print("TEMP. ACUARIO.:");  // Texto del SMS
      Serial2.println(temp1);
      Serial2.print("TEMP. EXTERNA:");  // Texto del SMS
      Serial2.println(temp2);
      Serial2.print("PH:");  // Texto del SMS
      Serial2.println(phValue);
      delay(100);
      Serial2.println((char)26); //Comando de finalización ^Z
      delay(3000);
      inicializa();
      // Esperamos un tiempo para que envíe el SMS
    }
```

84. FIGURA 3.33. CODIGO QUE ENVIA VALORES DE LA TEMPERATURA Y HUMEDAD POR SMS (COMANDO SMS: 1234V) (PROPIO)

```
//****PROGRAMAR O ALMACENAR EL NUMERO DE CELULAR QUE RECIBIRA LAS ALARMAS*****************************
if ((letras[9]=='X')&&(c==0)){ //COMANDO 951551591X
    numero=clave.substring(0);
    numero.toCharArray(charBuf, 10);
    n=9;
    posicion=80;
    for (k=0; k<=n; k++) // se lee caracter (valor asccii) por caracter guardado en memoria eemprom
        {
          EEPROM.write(posicion, charBuf[k]);
          posicion++; // se incrementa posición o dirección
          delay(50);
        }
    numero="";
    n=89;
    for (k=80; k<n; k++) // se lee caracter (valor asccii) por caracter guardado en memoria eemprom
        {
          incoming_char =EEPROM.read(k);// se lee memoria y se guarda en la variable valorl
          numero = numero + incoming_char;
          delay(50);
        }
    inicializa();
    }
```

85. FIGURA 3.34. CODIGO QUE PROGRAMA DE CELULAR PARA ALARMAS POR SMS (COMANDO SMS: 951551591X) (PROPIO)

```
//*******PROGRAMAR SET POINT DE TEMPERATURA POR SMS*****************************************
      if ((letras[2]=='T')&&(letras[3]=='X')){ //COMANDO 28T, SETPOINT DE TEMPERATURA
          tp=clave.substring(0);
          tp.toCharArray(charBuf, 3);
          ts=tp.toInt();
          inicializa();
          EEPROM.write(70, ts);
          tsmin=ts-0.5;
          tsmax=ts+0.5;
          talarma1=tsmin-2;
          talarma2=tsmax+1;
          delay(10);
        }
//*****************************************************************************************
//**************PROGRAMAR SET POINT DE NIVEL*************************************************
      if ((letras[2]=='N')&&(letras[3]=='X')){ // COMANDO 32N, SETPOINT DE NIVEL
          np=clave.substring(0);
          np.toCharArray(charBuf, 3);
          ns=np.toInt();
          inicializa();
          nsmin=ns-2;
          nsmax=ns+1;
          ns25=ns*0.25;
          ns25=ns-ns25;
          ns50=ns*0.5;
          ns50=ns-ns50;
          ns75=ns*0.75;
          ns75=ns-ns75;
          //nalarma1=nsmin-2;
```

86. FIGURA 3.35. CODIGO QUE PROGRAMA SET POIT DE TEMPERATURA Y NIVEL POR SMS (COMANDO SMS: 18TX, 30NX) (PROPIO)

```
//*****************************************************************************************
//COMUNICACION UNSADO PUERTO SERIAL /BLUETHOTH
//*****************************************************************************************
void serial_bluetooth(){
  while(Serial1.available()>0){
        clave1 = Serial1.read();
        clave += clave1;
  }
  clave.toCharArray(letras, 12);
  c=0;
      if ( letras[0] == 'A' ){//ABRE bomba_tanque  COMANDO : A
          digitalWrite(bomba_tanque, LOW);
          inicializa();
        }
      if ( letras[0] == 'a' ){// CIERRA bomba_tanque COMANDO : a
          digitalWrite(bomba_tanque, HIGH);
          inicializa();
        }

      if ( letras[0] == 'B' )//ENCIENDE TERMOSTATO  COMANDO : B
        {
          digitalWrite(termostato, LOW);
          inicializa();
        }

      if ( letras[0] == 'b' )//APAGA TERMOSTATO  COMANDO : b
        {
          digitalWrite(termostato, HIGH);
          inicializa();
```

87. FIGURA 3.36. CODIGO PARA CONTROL Y SUPERVISION POR PUERTO PARA MODULO BLUETOOTH SIMILAR AL GSM (PROPIO)

```
//*****************************************************************************
if ( letras[0] == 'Q' )// ENVIA POR BLUETTOTH VALORES SETEADOS DE VARIABLES COMANDO : Q
  {
    Serial1.print("SET POINT TEMPERATURA ACUARIO:");
    Serial1.println(ts);
    Serial1.print("TEMP_min:");
    Serial1.println(tsmin);
    Serial1.print("TEMP_max:");
    Serial1.println(tsmax);
    Serial1.print("SET POINT NIVEL DE ACUARIO:");
    Serial1.println(ns);
    Serial1.print("NIVEL_min:");
    Serial1.println(nsmin);
    Serial1.print("NIVEL_max:");
    Serial1.println(nsmax);
    Serial1.print("NIVEL 25%:");
    Serial1.println(ns25);
    Serial1.print("NIVEL 50%:");
    Serial1.println(ns50);
    Serial1.print("NIVEL 75%:");
    Serial1.println(ns75);
    Serial1.print("SET POINT PH ACUARIO:");
    Serial1.println(ps);
    Serial1.print("PHmin:");
    Serial1.println(psmin);
    Serial1.print("PHmax:");
    Serial1.println(psmax);
    inicializa();
```

88. FIGURA 3.37. CODIGO QUE ENVIA VALORES SETEADOS POR PUERTO PARA MODULO BLUETOOTH (COMANDO: Q) (PROPIO)

```
//*****************************************************************************
void alarmas(){
if (alarma==0){
  if (( nivel_ok <= nalarmal ) or ( templ <= talarmal )) {
    Serial2.print("AT+CMGS="); //Numero al que vamos a enviar el mensaje
    Serial2.print((char)34);
    Serial2.print(numero1);
    Serial2.println((char)34);
    delay(1000);
    Serial2.println("Alarma de nivel bajo o temperatura baja... !!!" );
    Serial2.print("NIVEL:");  // Texto del SMS
    Serial2.println(nivel_ok);
    Serial2.print("TEMP. ACUARIO.:");  // Texto del SMS
    Serial2.println(templ);
    delay(1000);
    Serial2.println((char)26); //Comando de finalización ^Z
    delay(1000);
    Serial2.println();
    delay(5000);  // Esperamos un tiempo para que envíe el SMS
    inicializa();
    alarma=1;
  }
```

89. FIGURA 3.38. CODIGO PARA ACTIVACION DE ALARMAS (PROPIO).

3.5. PRUEBAS Y RESULTADOS.

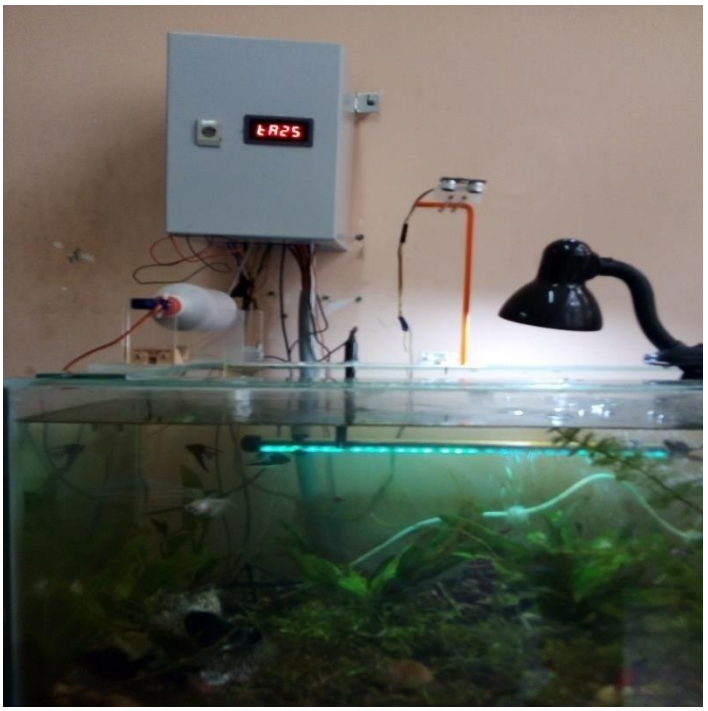
3.5.1. FOTOS DEL PROYECTO

90.FIGURA 3.39. FOTO DE SISTEMA DE ACUARIO AUTOMATIZADO CON TECNOLOGIA GPRS Y BLUETOOTH (PROPIO)

91.FIGURA 3.40 FOTO DE ACUARIO CON LUZ DE DIA (PROPIO)

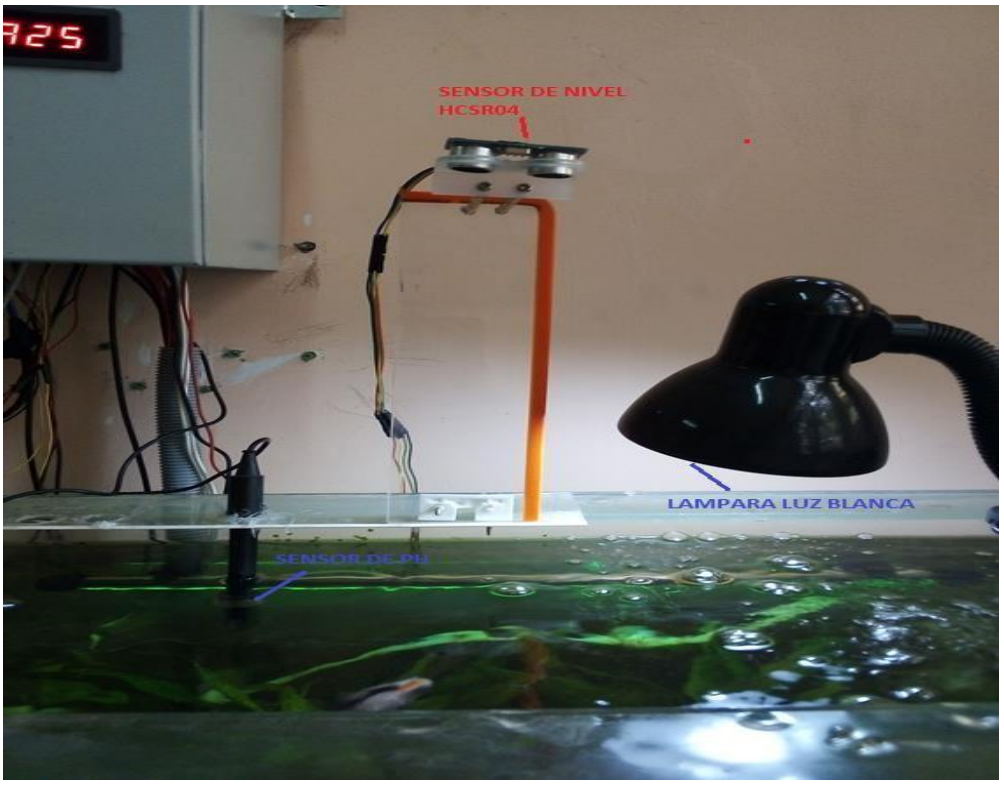
92.FIGURA 3.41. FOTO DE ACUARIO CON LUZ DE NOCHE (PROPIO)

93. FIGURA 3.42. FOTO DE ACUARIO CON TANQUE DE ALMACEN DE AGUA (PROPIO)

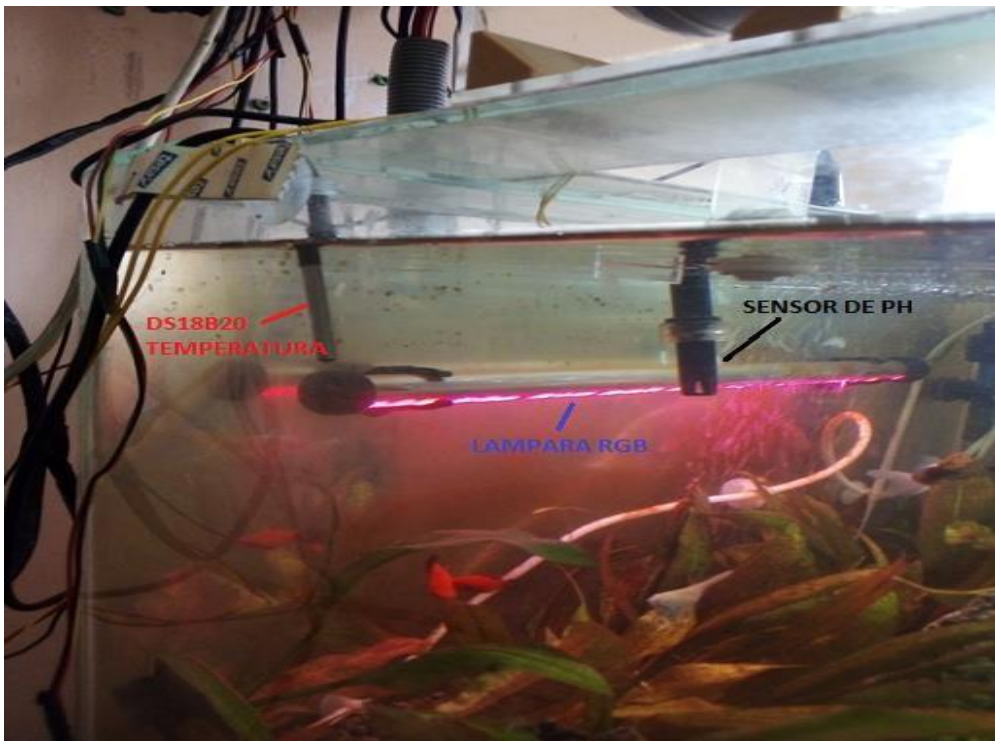


94.FIGURA 3.43. FOTO DE VISTA EXTERNA DE EQUIPO DE CONTROL Y SUPERVISION DE ACUARIO (PROPIO)

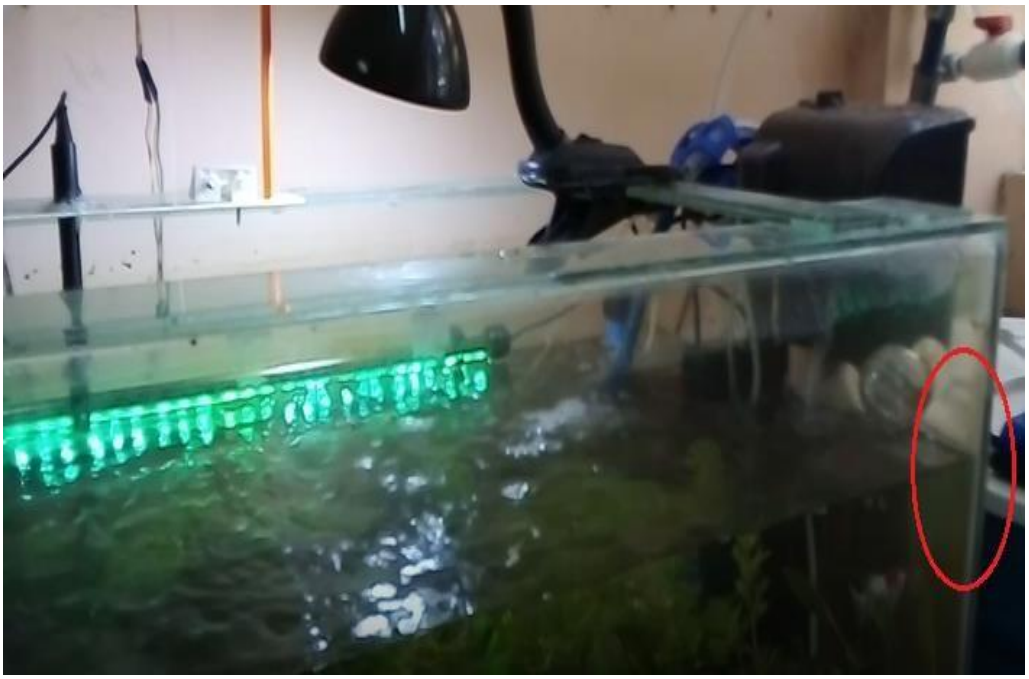
95.FIGURA 3.44. FOTO DE VISTA DE EQUIPOS EN ACUARIO (PROPIO)



96.FIGURA 3.45. FOTO DE VISTA DE SENSOR DE NIVEL, PH Y LAMPARA DE LUZ BLANCA (PROPIO)



97.FIGURA 3.46.FOTO DE VISTA DE SENSOR DE PH, TEMPERATURA Y LAMPARA RGB AL INTERIOR DEL ACUARIO (PROPIO)

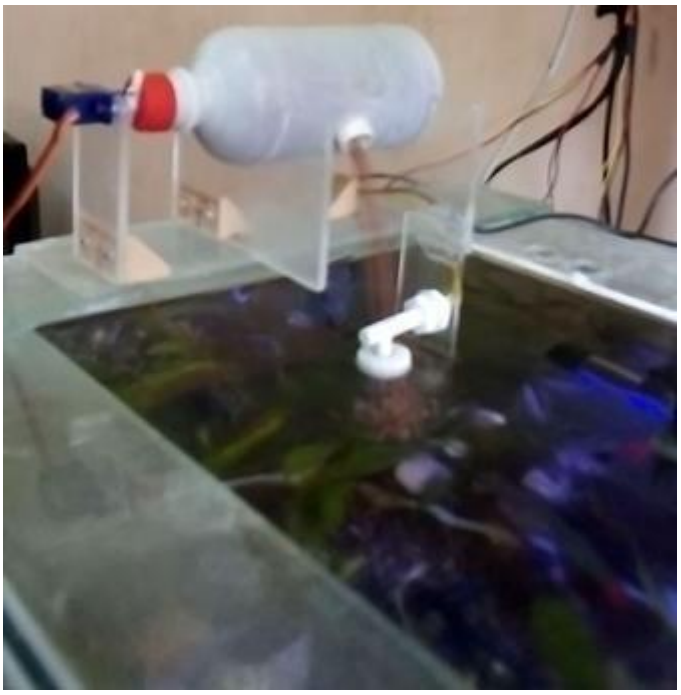


98.FIGURA 3.47. FOTO DE NIVEL DEL AGUA BAJADO AL 25% DEL TOTAL DEL ACUARIO (PROPIO)

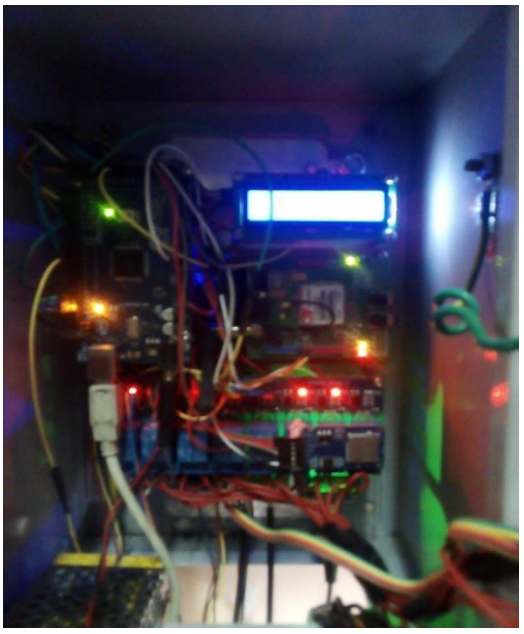
99.FIGURA 3.48. FOTO DE LLENANDO DE AGUA EL ACUARIO (PROPIO)

100.FIGURA 3.49. FOTO DE LLENANDO DE AGUA EL ACUARIO (PROPIO)

101.FIGURA 3.50. FOTO DE DISPENSADOR DE COMIDA PARA PECES (PROPIO)



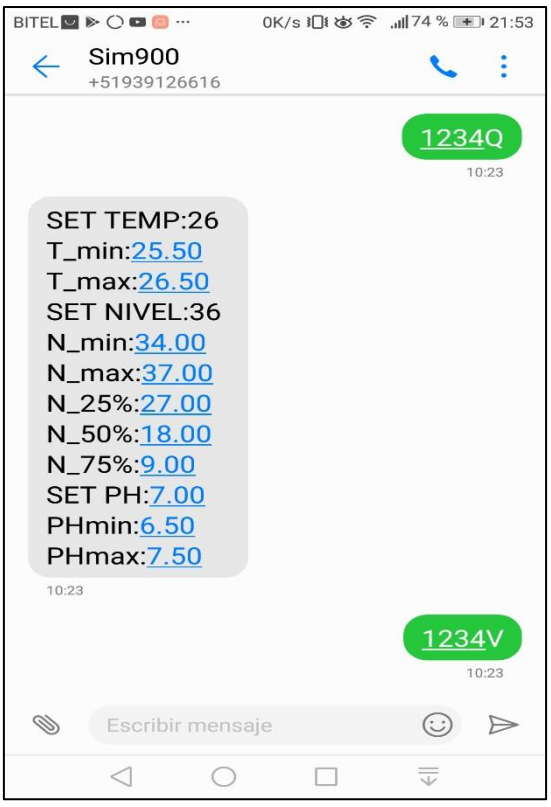
102.FIGURA 3.51. FOTO DE DISPENSADOR DE COMIDA PARA PECES CONTROLADA POR SERVO MOTOR (PROPIO)



103.FIGURA 3.52. FOTO DE TARJETA ELECTRONICA DE CONTROL Y SUPERVISON DE ACUARIOS (PROPIO)

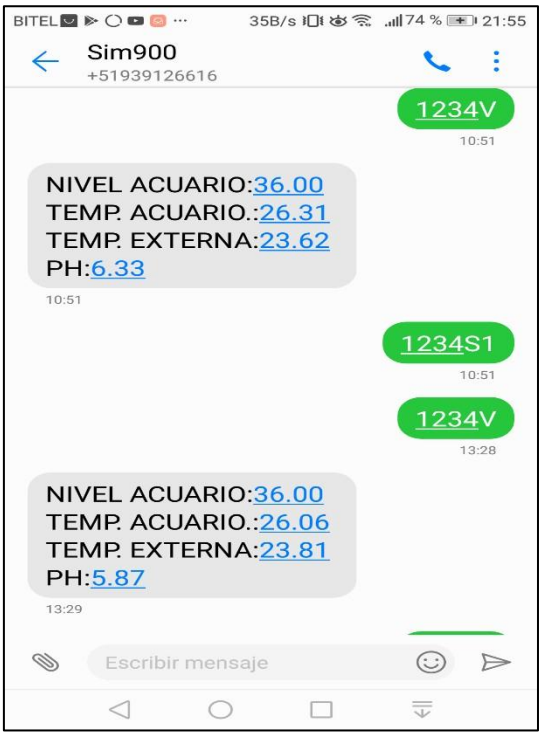
104.FIGURA 3.53. FOTO DE LA CAJA QUE CONTIENE TARJETA ELECTRONICA (PROPIO)

3.5.2. CAPTURA DE PANTALLAS DE CONTROL Y SUPERVISION POR SMS



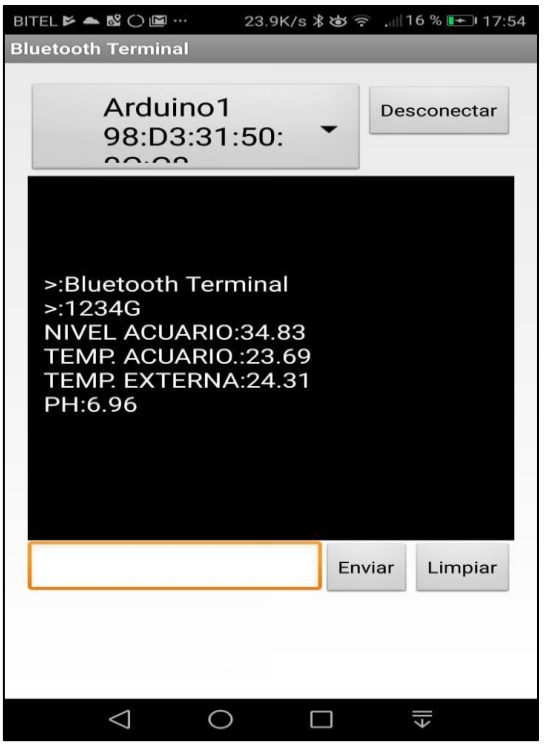
105. FIGURA 3.54. MENSAJE RECIBIDO DE VALORES SETEADOS COMO RESPUESTA AL COMANDO "1234Q". (PROPIO)

106. FIGURA 3.55. MENSAJE RECIBIDO DE ESTADO DE EQUIPOS Y VALOR DE VARIABLES COMO RESPUESTA AL COMANDO "12348" Y "1234V" (PROPIO)

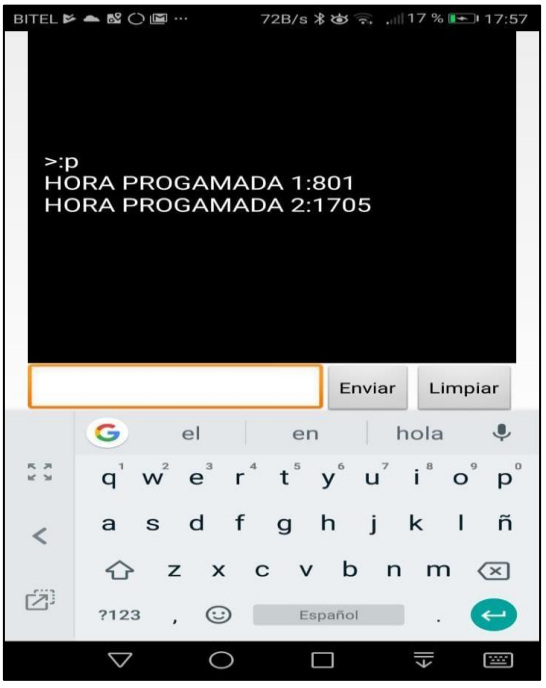


107.FIGURA 3.56. MENSAJE RECIBIDO DE VALORES DE VARIABLES COMO RESPUESTA AL COMANDO "1234V". (PROPIO)

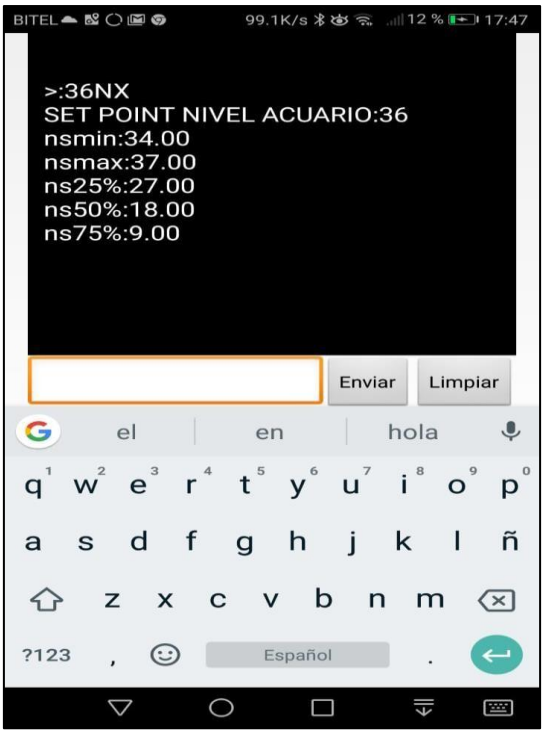
3.5.3. CAPTURA DE PANTALLAS DE CONTROL Y SUPERVISION POR BLUETOOTH.



108.FIGURA 3.57. MENSAJE RECIBIDO DE VALORES DE VARIABLES COMO RESPUESTA AL COMANDO "1234G". (PROPIO)



109.FIGURA 3.58. MENSAJE RECIBIDO DE HORAS PROGRAMADAS DE COMIDAS COMO RESPUESTA AL COMANDO "p". (PROPIO)



110.FIGURA 3.59. PROGRAMANDO LA TEMPERATURA DE REFERENCIA A 36° CON COMANDO "36NX". (PROPIO)

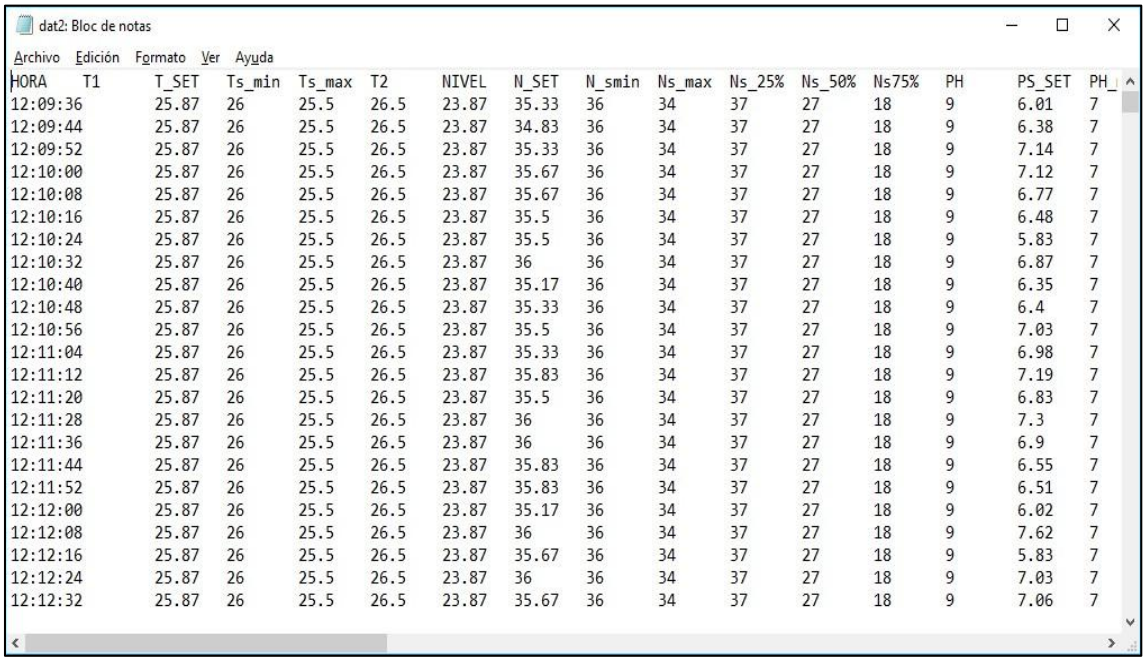
111.FIGURA 3.60. MENSAJE RECIBIDO DE ESTADO DE EQUIPOS COMO RESPUESTA AL COMANDO "1234E". (PROPIO)

3.5.4. CAPTURA DE DATOS Y GRAFICAS DE LOS EVENTOS EN EL ACUARIO.

Para el análisis del funcionamiento del acuario se procede hacer las capturas de los datos enviados por la tarjeta implementada. Los datos que se envía son:

- Temperatura de ambiente
- Temperatura del acuario
- Nivel de agua en acuario en cm
- Valor del PH del agua del acuario
- Valores de referencia de temperatura, nivel, PH
- Valores mínimos y máximos de temperatura, nivel y PH
- Estados de los equipos del acuario : Bombas, calentador, filtros, lámparas

Los datos son almacenados en una memoria SD en un archivo tipo texto (txt).



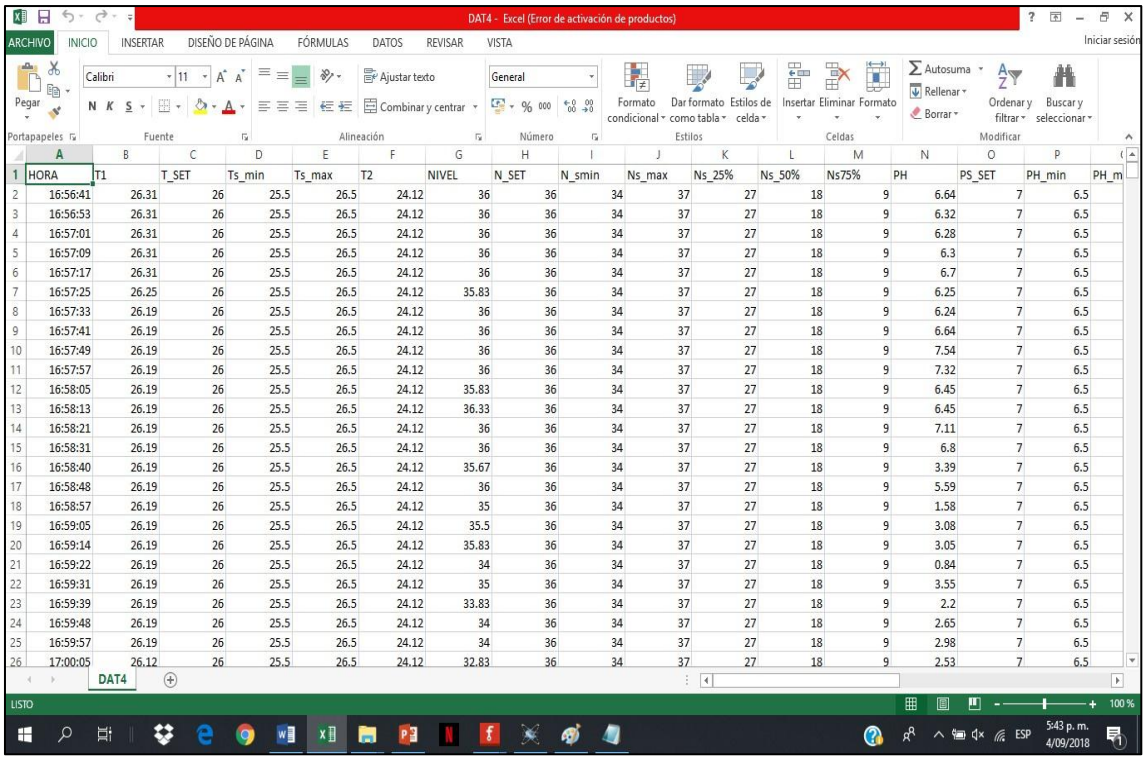
dat2: Bloc de notas

Archivo Edición Formato Ver Ayuda

HORA	T1	T_SET	Ts_min	Ts_max	T2	NIVEL	N_SET	N_smin	Ns_max	Ns_25%	Ns_50%	Ns75%	PH	PS_SET	PH_
12:09:36		25.87	26	25.5	26.5	23.87	35.33	36	34	37	27	18	9	6.01	7
12:09:44		25.87	26	25.5	26.5	23.87	34.83	36	34	37	27	18	9	6.38	7
12:09:52		25.87	26	25.5	26.5	23.87	35.33	36	34	37	27	18	9	7.14	7
12:10:00		25.87	26	25.5	26.5	23.87	35.67	36	34	37	27	18	9	7.12	7
12:10:08		25.87	26	25.5	26.5	23.87	35.67	36	34	37	27	18	9	6.77	7
12:10:16		25.87	26	25.5	26.5	23.87	35.5	36	34	37	27	18	9	6.48	7
12:10:24		25.87	26	25.5	26.5	23.87	35.5	36	34	37	27	18	9	5.83	7
12:10:32		25.87	26	25.5	26.5	23.87	36	36	34	37	27	18	9	6.87	7
12:10:40		25.87	26	25.5	26.5	23.87	35.17	36	34	37	27	18	9	6.35	7
12:10:48		25.87	26	25.5	26.5	23.87	35.33	36	34	37	27	18	9	6.4	7
12:10:56		25.87	26	25.5	26.5	23.87	35.5	36	34	37	27	18	9	7.03	7
12:11:04		25.87	26	25.5	26.5	23.87	35.33	36	34	37	27	18	9	6.98	7
12:11:12		25.87	26	25.5	26.5	23.87	35.83	36	34	37	27	18	9	7.19	7
12:11:20		25.87	26	25.5	26.5	23.87	35.5	36	34	37	27	18	9	6.83	7
12:11:28		25.87	26	25.5	26.5	23.87	36	36	34	37	27	18	9	7.3	7
12:11:36		25.87	26	25.5	26.5	23.87	36	36	34	37	27	18	9	6.9	7
12:11:44		25.87	26	25.5	26.5	23.87	35.83	36	34	37	27	18	9	6.55	7
12:11:52		25.87	26	25.5	26.5	23.87	35.83	36	34	37	27	18	9	6.51	7
12:12:00		25.87	26	25.5	26.5	23.87	35.17	36	34	37	27	18	9	6.02	7
12:12:08		25.87	26	25.5	26.5	23.87	36	36	34	37	27	18	9	7.62	7
12:12:16		25.87	26	25.5	26.5	23.87	35.67	36	34	37	27	18	9	5.83	7
12:12:24		25.87	26	25.5	26.5	23.87	36	36	34	37	27	18	9	7.03	7
12:12:32		25.87	26	25.5	26.5	23.87	35.67	36	34	37	27	18	9	7.06	7

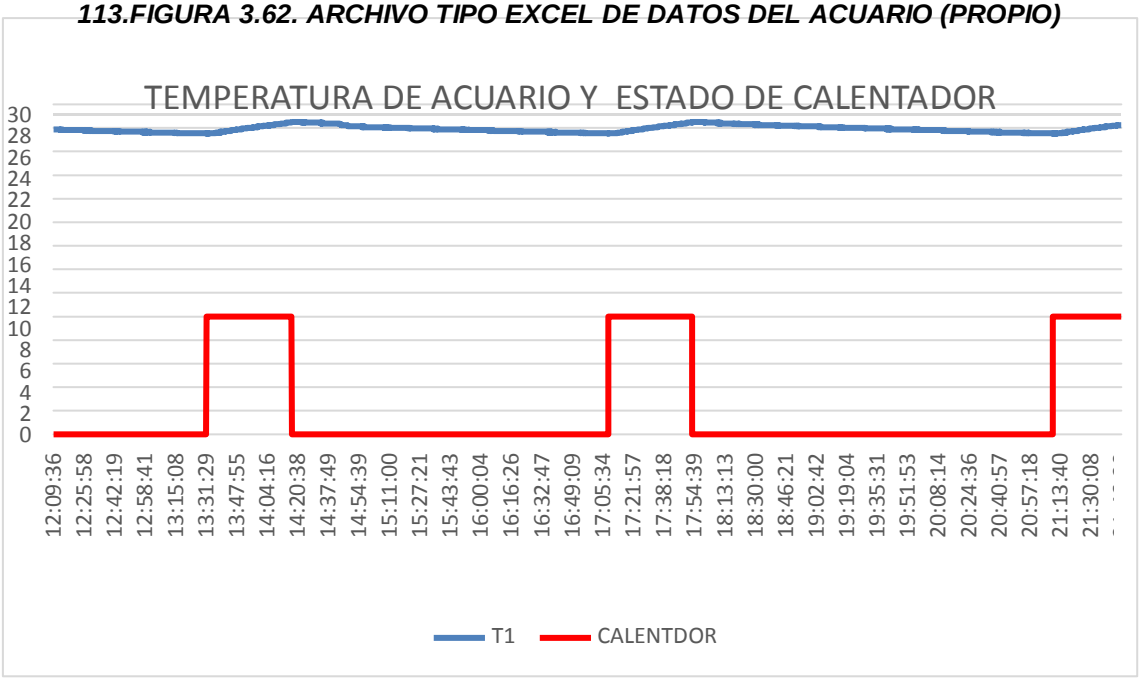
112.FIGURA 3.61. ARCHIVO TIPO TXT DE DATOS DEL ACUARIO (PROPIO)

Para realizar la gráficas de los datos se utiliza el software EXCEL para abrir el archivo “txt”, luego se procede a realizar las gráficas como se muestra a continuación.



HORA	T1	T_SET	Ts_min	Ts_max	T2	NIVEL	N_SET	N_smin	Ns_max	Ns_25%	Ns_50%	Ns75%	PH	PS_SET	PH_min	PH_m
16:56:41	26.31	26	25.5	26.5	24.12	36	36	34	37	27	18	9	6.64	7	6.5	
16:56:53	26.31	26	25.5	26.5	24.12	36	36	34	37	27	18	9	6.32	7	6.5	
16:57:01	26.31	26	25.5	26.5	24.12	36	36	34	37	27	18	9	6.28	7	6.5	
16:57:09	26.31	26	25.5	26.5	24.12	36	36	34	37	27	18	9	6.3	7	6.5	
16:57:17	26.31	26	25.5	26.5	24.12	36	36	34	37	27	18	9	6.7	7	6.5	
16:57:25	26.25	26	25.5	26.5	24.12	35.83	36	34	37	27	18	9	6.25	7	6.5	
16:57:33	26.19	26	25.5	26.5	24.12	36	36	34	37	27	18	9	6.24	7	6.5	
16:57:41	26.19	26	25.5	26.5	24.12	36	36	34	37	27	18	9	6.64	7	6.5	
16:57:49	26.19	26	25.5	26.5	24.12	36	36	34	37	27	18	9	7.54	7	6.5	
16:57:57	26.19	26	25.5	26.5	24.12	36	36	34	37	27	18	9	7.32	7	6.5	
16:58:05	26.19	26	25.5	26.5	24.12	35.83	36	34	37	27	18	9	6.45	7	6.5	
16:58:13	26.19	26	25.5	26.5	24.12	36.33	36	34	37	27	18	9	6.45	7	6.5	
16:58:21	26.19	26	25.5	26.5	24.12	36	36	34	37	27	18	9	7.11	7	6.5	
16:58:31	26.19	26	25.5	26.5	24.12	36	36	34	37	27	18	9	6.8	7	6.5	
16:58:40	26.19	26	25.5	26.5	24.12	35.67	36	34	37	27	18	9	3.39	7	6.5	
16:58:48	26.19	26	25.5	26.5	24.12	36	36	34	37	27	18	9	5.59	7	6.5	
16:58:57	26.19	26	25.5	26.5	24.12	35	36	34	37	27	18	9	1.58	7	6.5	
16:59:05	26.19	26	25.5	26.5	24.12	35.5	36	34	37	27	18	9	3.08	7	6.5	
16:59:14	26.19	26	25.5	26.5	24.12	35.83	36	34	37	27	18	9	3.05	7	6.5	
16:59:22	26.19	26	25.5	26.5	24.12	34	36	34	37	27	18	9	0.84	7	6.5	
16:59:31	26.19	26	25.5	26.5	24.12	35	36	34	37	27	18	9	3.55	7	6.5	
16:59:39	26.19	26	25.5	26.5	24.12	33.83	36	34	37	27	18	9	2.2	7	6.5	
16:59:48	26.19	26	25.5	26.5	24.12	34	36	34	37	27	18	9	2.65	7	6.5	
16:59:57	26.19	26	25.5	26.5	24.12	34	36	34	37	27	18	9	2.98	7	6.5	
17:00:05	26.12	26	25.5	26.5	24.12	32.83	36	34	37	27	18	9	2.53	7	6.5	

113.FIGURA 3.62. ARCHIVO TIPO EXCEL DE DATOS DEL ACUARIO (PROPIO)

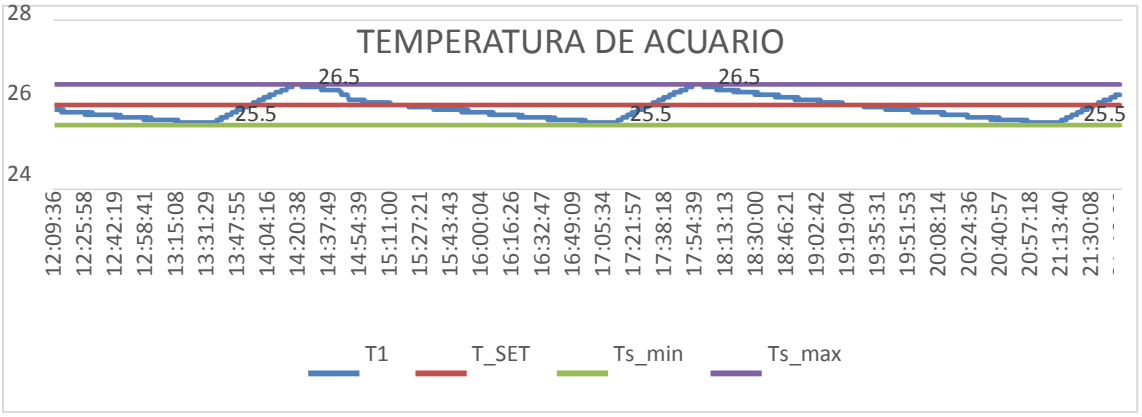


114.FIGURA 3.63. GRAFICA DE LA TEMPERATURA Y CALENTADOR DEL ACUARIO (PROPIO)

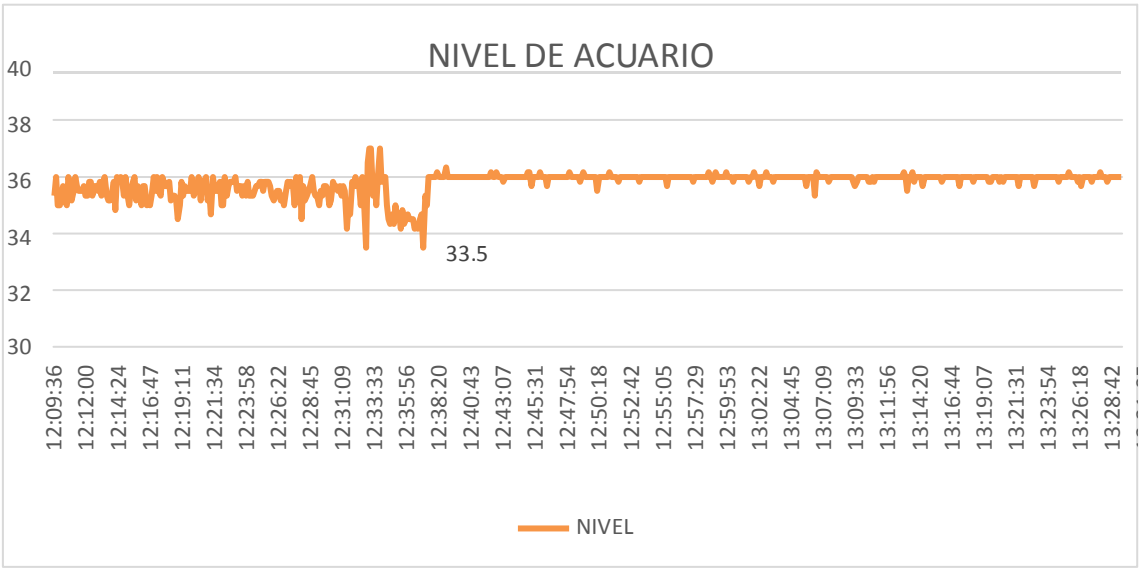
En la Figura 3.63 de la Temperatura de Acuario y Estado del Calentador, y la Figura 3.64 se observa que el calentador se enciende cuando la temperatura está en 25.5°C y se apaga cuando alcanza la temperatura de 26.5°. El valor

referencial es 26°C y por programa se ha establecido una banda de 1° C el cual se cumple al observar la gráfica.

En estas graficas también se observa que para alcanzar la temperatura de 26.5°C iniciando en 25.5°C tiene una demora de 48 minutos aproximadamente, y para bajar de 26.5°C a 25.5°C tiene una de mora de 3 horas aproximadamente.

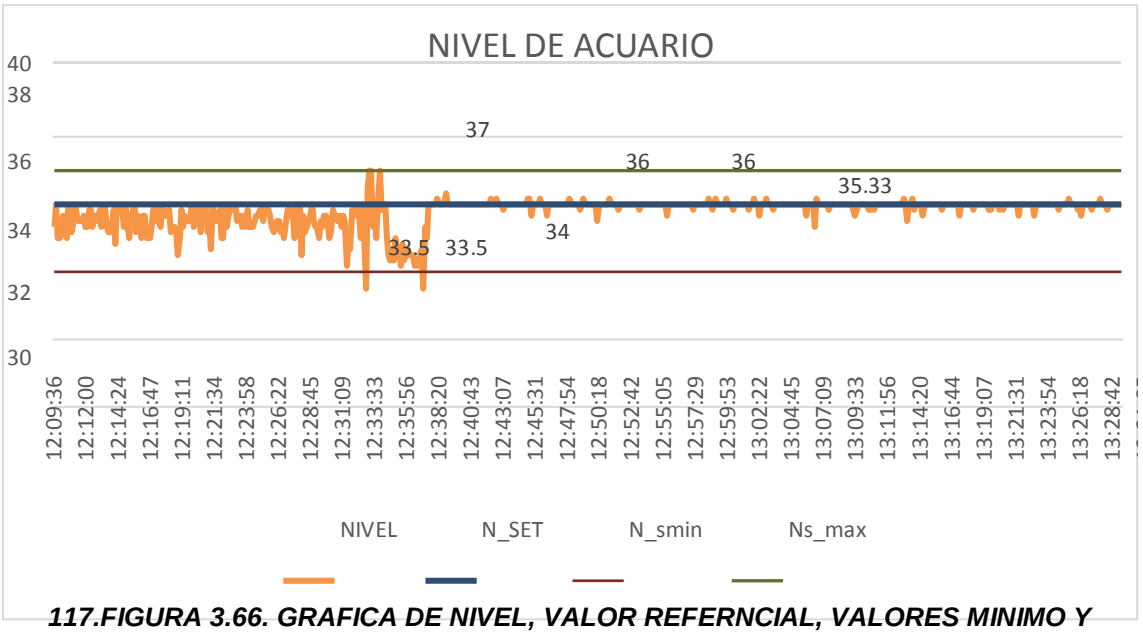


115.FIGURA 3.64. GRAFICA DE LA TEMPERATURA DEL ACUARIO CON UNA BANDA DE 1°C (PROPIO)



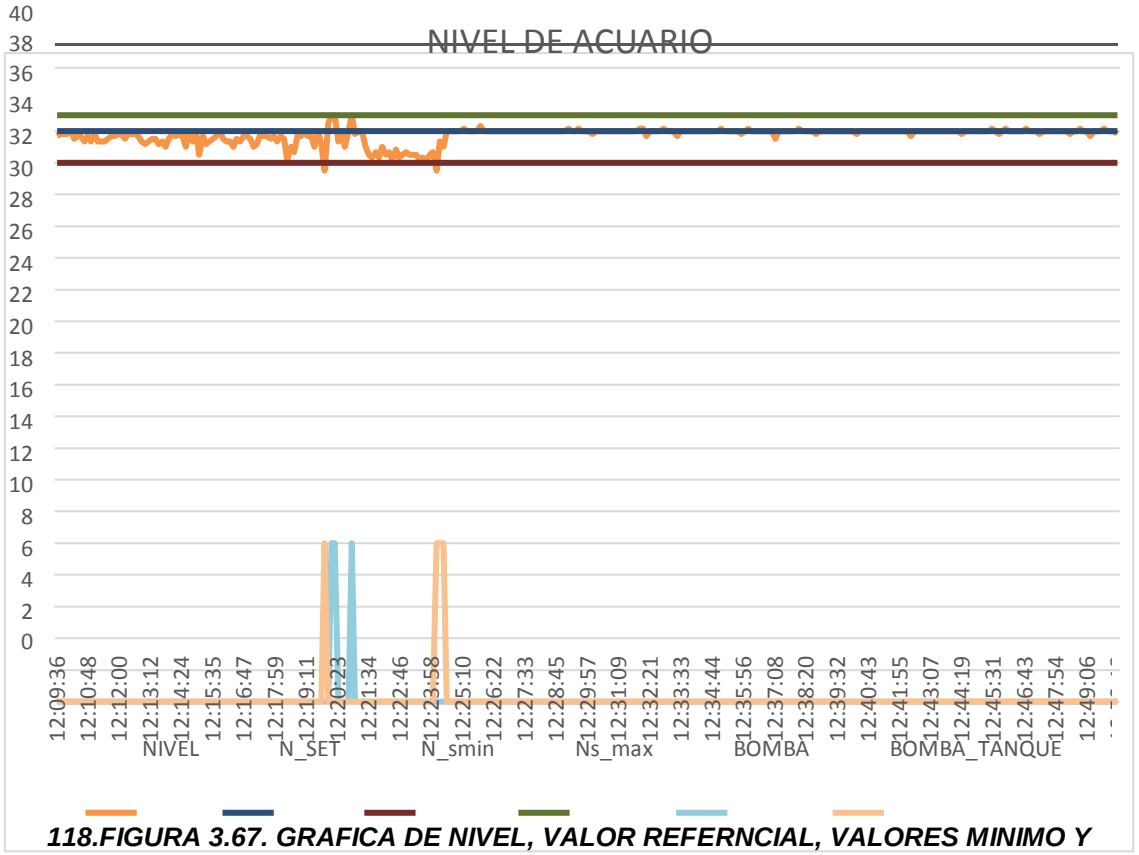
116.FIGURA 3.65. GRAFICA DE NIVEL ACUARIO (PROPIO)

En la Figura 3.65. y 3.66 donde se gráfica el Nivel, se observa que se está haciendo un cambio de agua al 10% ya que el nivel inferior es de 33.5 cm y luego se alcanza el valor referencial de 36cm. Este cambio demoro aproximadamente 20 minutos. Empezó 12:09 y acabo 12:38

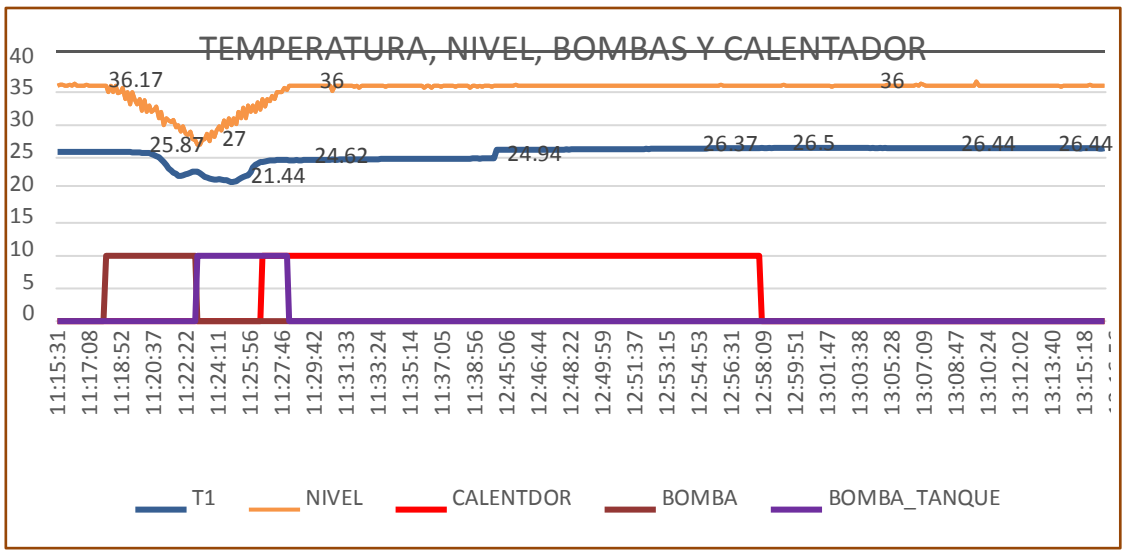


117.FIGURA 3.66. GRAFICA DE NIVEL, VALOR REFERNCIAL, VALORES MINIMO Y MAXIMO (PROPIO)

En la Figura 3.67 se observa que se activa la Bomba de Agua para bajar el nivel (color celeste) y se activa la bomba de tanque (color rosado) para subir nivel. El tiempo que se demora de 34.5cm hasta llegar a 36cm es de aproximadamente 3 minutos.

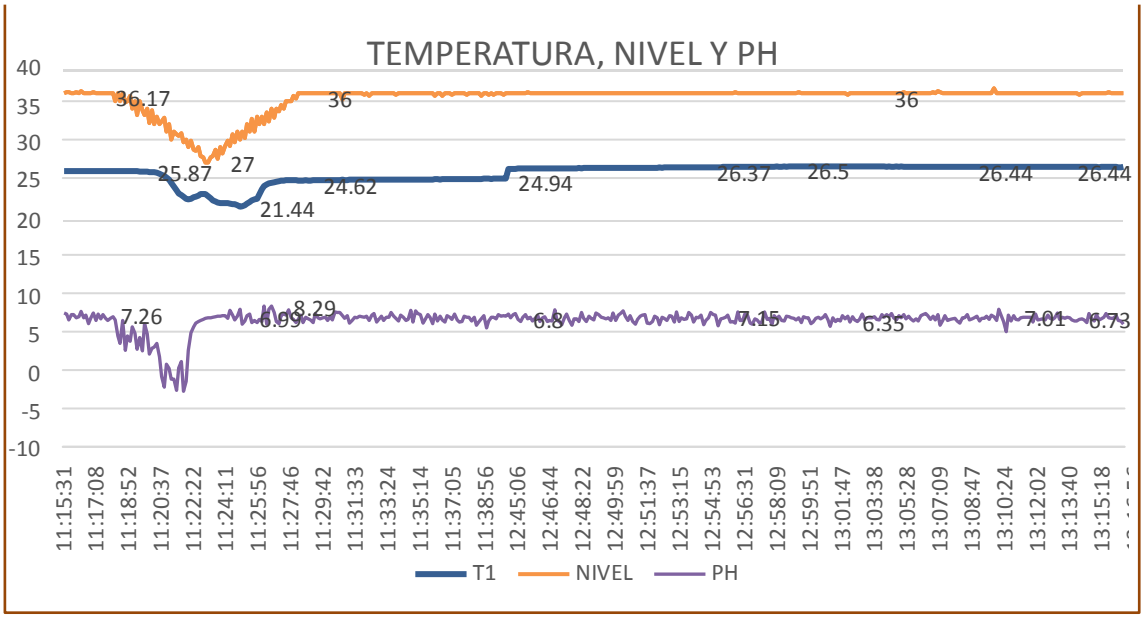


118.FIGURA 3.67. GRAFICA DE NIVEL, VALOR REFERNCIAL, VALORES MINIMO Y MAXIM Y ESTADO DE BOMBAS (PROPIO)



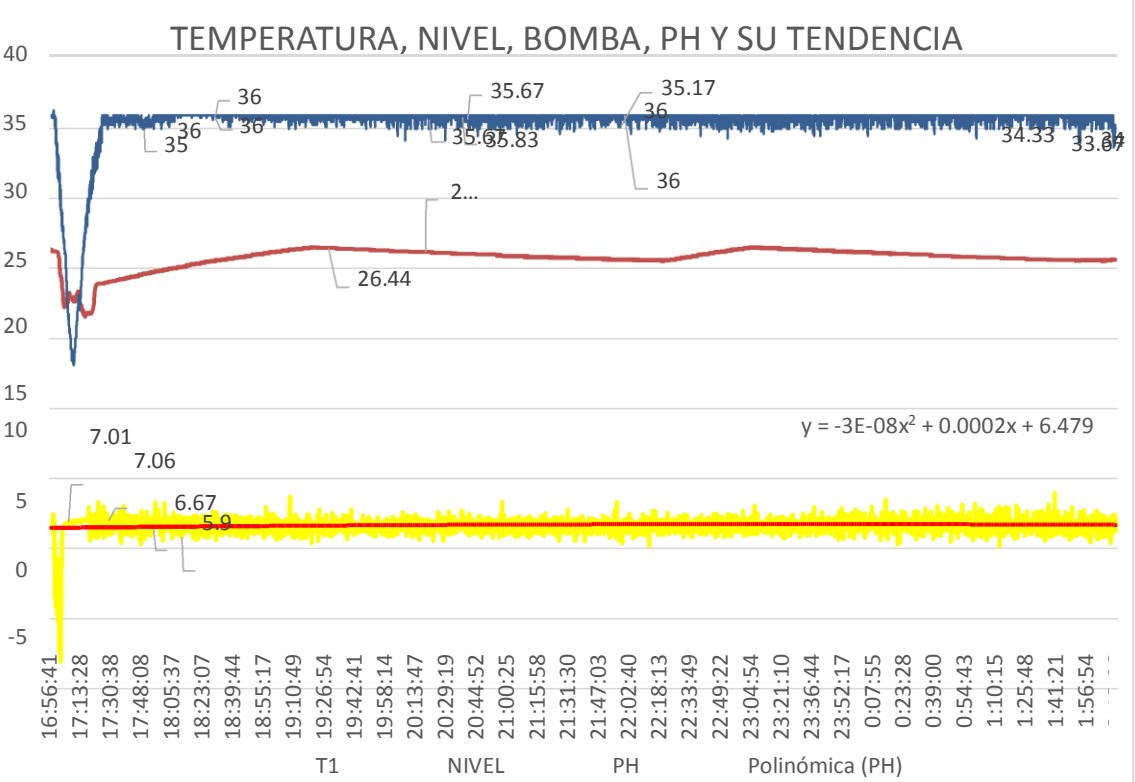
119.FIGURA 3.68. GRAFICA DE NIVEL, TEMPERATURA, ESTADO DE BOMBAS Y CALENTADOR (PROPIO)

En la Figura 3.68 se observa la influencia de cambio de nivel de agua sobre la Temperatura, lo cual se comporta como una perturbación, luego se ve que la temperatura se recupera y se mantiene en los límites establecidos entre 25.5°C y 26.5 °C.



120.FIGURA 3.69. GRAFICA DE NIVEL, TEMPERATURA, PH (PROPIO)

En la Figura 3.69 se observa la influencia de cambio de nivel de agua sobre la Temperatura, y el PH del agua, lo cual se comporta como una perturbación, luego se ve que la temperatura se recupera y se mantiene en los límites establecidos entre 25.5°C y 26.5 °C, de igual manera para el PH el cual se mantiene entre 6 y 7.5.



121.FIGURA 3.70. GRAFICA DE NIVEL, TEMPERATURA, PH Y POLINOMICA DE PH (PROPIO)

En la Figura 3.70 a la gráfica del PH se le ha determinado la tendencia la cual está alrededor de 7.

CAPITULO IV

COSTOS DEL PROYECTO

4.1. COSTOS DEL PROYECTO

En este capítulo se mostrara la tabla de costos para la implementación del proyecto en una se mostrara solo los costos de materiales directos, materiales Indirectos, mano de obra directa, mano de obra indirecta, gastos generales, costos de materiales directos e indirectos y costos totales.

Tabla N°5. MATERIALES DIRECTOS (PROPIO)

MATERIALES DIRECTOS	
CANTIDAD	DESCRIPCION
1	ARDUINO MEGA 2560
2	SENSOR DE TEMPERATURA DS18B20B
1	SENSOR DE NIVEL POR ULTRASONIDO HCSR04
1	SENSOR DE PH
1	RTC DS1307
1	FUENTE ENT.12V 2A
8	RELES 5V / 220VAC 10A
1	LCD 16X2 I2C
1	DISPLAY 4 DIGITOS
1	MODULO GSM/GPRS SIM900
1	MODULO BLUETOOTH
1	SERVO MOTOR
1	PLACA DE CIRCUITO IMPRESO
1	CAJA PARA PLACA IMPRESA
1	CONECTORES
1	FILTRO DE CABEZA 800L/H
1	FILTRO TIPO CASCADA 800L/H
1	BOMBA DE AIRE
1	BOMBA PARA ACUARIO
1	BOMBA PARA TANQUE
1	LAMPARA LUZ BLANCA
1	LAMPARA LUZ AZUL
1	LAMPARA RGB
1	CALENTADOR
1	OTROS COMPONENTES VARIOS

Tabla N° 6. MATERIALES INDIRECTOS (INSUMOS) (PROPIO)

MATERIALES INDIRECTOS (INSUMOS)	
CANTIDAD	DESCRIPCION
1 Rollos	CABLE ELECTRICO #18 - CABLE AUTOMOTRIZ
2	CINTA AISLANTE
1 kg.	PEGAMENTO
3m	TUBO CONDUIT/PROTECCION DE CABLES
1 kg.	trapo industrial

Tabla N° 7. MANO DE OBRA DIRECTA (PROPIO)

MANO DE OBRA DIRECTA	
CANTIDAD	ESPECIALIDAD
1	Electricista
1	Electronico
1	Practicante

Tabla N° 8. MANO DE OBRA INDIRECTA (PROPIO)

MANO DE OBRA INDIRECTA	
CANTIDAD	ESPECIALIDAD
1	Supervisor
1	Prevencionista
1	Dibujante CAD

Tabla N° 9. GASTOS GENERALES (SERVICIOS A TODO COSTO) (PROPIO)

GASTOS GENERALES (SERVICIOS A TODO COSTO)	
CANTIDAD	ESPECIALIDAD
1	SERVICIO DE INSTALACION PUNTOS ELECTRICOS
2 dias	ENERGIA
	INTERNET
1	VIATICOS
	AGUA POTABLE
	TRANSPORTE
	GASTOS DE OFICINA

Tabla N° 10. COSTO TOTAL DE MATERIALES (PROPIO)

COSTO DE MATERIALES DIRECTOS (MATERIA PRIMA)				
ITEM	DESCRIPCION	CANTIDAD	COSTO UNITARIO	COSTO TOTAL
1	ARDUINO MEGA 2560	1	S/. 65.00	S/. 65.00
2	SENSOR DE TEMPERATURA DS18B20B	2	S/. 15.00	S/. 30.00
3	SENSOR DE NIVEL POR ULTRASONIDO HCSR04	1	S/. 15.00	S/. 15.00
4	SENSOR DE PH	1	S/. 180.00	S/. 180.00
5	RTC DS1307	1	S/. 14.00	S/. 14.00
6	FUENTE ENT.12V 2A	1	S/. 45.00	S/. 45.00
7	RELES 5V / 220VAC 10A	8	S/. 15.00	S/. 120.00
8	LCD 16X2 I2C	1	S/. 25.00	S/. 25.00
9	DISPLAY 4 DIGITOS	1	S/. 50.00	S/. 50.00
10	MODULO GSM/GPRS SIM900	1	S/. 160.00	S/. 160.00
11	MODULO BLUETOOTH	1	S/. 40.00	S/. 40.00
12	SERVO MOTOR	1	S/. 40.00	S/. 40.00
13	PLACA DE CIRCUITO IMPRESO	1	S/. 105.00	S/. 105.00
14	CAJA PARA PLACA IMPRESA	1	S/. 120.00	S/. 120.00
15	CONECTORES	1	S/. 50.00	S/. 50.00
16	FILTRO DE CABEZA 800L/H	1	S/. 80.00	S/. 80.00
17	FILTRO TIPO CASCADA 800L/H	1	S/. 100.00	S/. 100.00
18	BOMBA DE AIRE	1	S/. 50.00	S/. 50.00
19	BOMBA PARA ACUARIO	1	S/. 50.00	S/. 50.00
20	BOMBA PARA TANQUE	1	S/. 50.00	S/. 50.00
21	LAMPARA LUZ BLANCA	1	S/. 30.00	S/. 30.00
22	LAMPARA LUZ AZUL	1	S/. 25.00	S/. 25.00
23	LAMPARA RGB	1	S/. 60.00	S/. 60.00
24	CALENTADOR	1	S/. 80.00	S/. 80.00
25	OTROS COMPONENTES VARIOS	1	S/. 150.00	S/. 150.00
COSTO TOTAL MATERIALES DIRECTOS				S/. 1,734.00

COSTO DE MATERIALES INDIRECTOS (INSUMOS)				
ITEM	DESCRIPCION	CANTIDAD	COSTO UNITARIO	COSTO TOTAL
1	CABLE ELECTRICO #18	1	120	S/. 120.00
2	CINTA AISLANTE	2	5	S/. 10.00
3	PEGAMENTO	1	16	S/. 16.00
4	TUBO CONDUIT/PROTECCION DE CABLES (METROS)	3	4	S/. 12.00
5	TRAPO INDUSTRIAL (KG)	1	10	S/. 10.00
COSTO TOTAL MATERIALES INDIRECTOS				S/. 168.00

COTOS TOTAL DE MATERIALES	S/. 1,902.00

Tabla N° 11. COSTO TOTAL DE MANO DE OBRA (PROPIO)

COSTO DE MANO DE OBRA DIRECTA				
CANTIDAD	ESPECIALIDAD	TIEMPO DE TRABAJO HRA	COSTO HORA	TOTAL
1	Electricista	16.00	12.00	192.00
1	Electronico	50.00	25.00	1250.00
1	Practicantes	30.00	6.00	180.00
	TOTAL	96.00		**1622.00**

COSTO DE MANO DE OBRA INDIRECTA				
CANTIDAD	ESPECIALIDAD	TIEMPO DE TRABAJO HRA	COSTO HORA	TOTAL
1	Ingeniero de Inspección	4.00	40.00	160.00
1	Prevencionista	2.00	12.00	24.00
1	Dibujante CAD	2.00	20.00	40.00
	TOTAL	8.00		**224.00**

COTOS TOTAL DE MANO DE OBRA	S/. 1,846.00

Tabla N° 12. COSTO TOTAL DE GASTOS GENERALES (PROPIO)

GASTOS GENERALES (EQUIPOS, HERRAMIENTAS, MAQUINAS, SERVICIOS)				
CANTIDAD	DENIOMINACION (Alquiler)	DIAS/horas TRABAJO	COSTO DIARIO	COSTO TOTAL
1	SERVICIO DE INSTALACION PUNTOS ELECTRICOS	1	80.00	80.00
1 KW/h	Energia Electrica (2 dias - 16 horas)	8	0.70	5.60
	Agua Potable			20.00
	Transporte			50.00
	Gastos de Oficina			120.00
TOTAL GASTOS GENERALES				S/. 275.60

Tabla N° 13. COSTO TOTAL DEL PROYECTO PROPUESTO (PROPIO)

COSTO TOTAL DEL PROYECTO PROPUESTO	
DESCRIPCION	COSTO
COTOS TOTAL DE MATERIALES (MD + MI)	1902.00
COTOS TOTAL DE MANO DE OBRA (MOD + MOI)	1846.00
TOTAL GASTOS GENERALES (Herramientas + Equipos + Maquinas + Servicios)	275.60
TOTAL DEL PROYECTO	S/. 4,023.60

CONCLUSIONES

- Se logró diseñar y construir un sistema automatizado para control y supervisión de un acuario usando tecnologías inalámbricas GPRS y BLUETOOTH
- Se logró determinar los bloques del sistema a implementar.
- Se logró seleccionar cada uno de los componentes como sensores, y demás partes para el correcto funcionamiento del sistema automatizado de control y supervisión de un acuario con tecnologías BLUETOOTH y GPRS.
- Se logró monitorear la temperatura nivel y PH en el Acuario
- Se logró Automatizar el sistema de iluminación, aireación, filtraje entre otros elementos en el acuario.
- Se logró implementar de manera eficiente un control de temperatura y nivel de un acuario.
- Se logró definir las alarmas para nivel, temperatura y PH, las cuales se transmiten por mensajes de texto.
- Se logró desarrollar el programa para todo el sistema electrónico, para la tarjeta ARDUINO MEGA 2560 para que se pueda comunicar serial mente con Modem SIM900 y por puerto serial con modulo BLUETOOTH HC06.
- Se logra establecer un protocolo de comunicación vía mensajes de texto entre la tarjeta electrónica desarrollada y el celular o dispositivo móvil.
- Se logra establecer un protocolo de comunicación vía comandos enviados por el puerto serial al módulo BLUETOOTH y hacia un móvil o celular.
- Se logró realizar pruebas al sistema y se comprobó el buen funcionamiento de todo el sistema.

BIBLIOGRAFÍA

(2009). Obtenido de learn.mikrooe.com: https://learn.mikroe.com/ebooks/microcontroladorespicc/

ABCelectronica.net. (s.f.). *ABCelectronica.net*. Obtenido de https://www.abcelectronica.net/: https://www.abcelectronica.net/productos/sensores/humedad/

ATMEL. (s.f.). Obtenido de http://www.atmel.com/devices/atmega32.aspx?tab=documents.(2)

BANCES, C. A. (2015). SISTEMA DE MONITOREO DE SEGURIDAD FÍSICA EN PLATAFORMA LIBRE DE COMPONENTES ELECTRÓNICOS PARA ASEGURAR LA GESTIÓN DE LOS NIVELES DE CONTINUIDAD DE LOS SERVICIOS INFORMÁTICOS EN LA CENTRAL DE DATOS USAT. *TESIS*. CHICLAYO, CHICLAYO, PERU.

Creus, A. (2013). Instrumentacion Industrial. En A. Creus, *Instrumentacion Industrial* (8va ed.). Madrid, España: Alfa y Omega.

Dadateca.unad.edu.co. (2014). Obtenido de http://datateca.unad.edu.co/contenidos/301120/2014_ii_reconocimiento_unidad2.pdf

Developers. (s.f.). *Developers*. Obtenido de http://developer.android.com/guide/index.html

Diego, D. P. (s.f.). *http://picmania.garcia-cuervo.net*. Obtenido de http://picmania.garcia-cuervo.net/recursos/redpictutorials/sensores/sensores_de_distancias_con_ultrasonidos.pdf

EL-KHOURI, N. (2016). ADAPTACION E IMPLEMENTACION DE UN SISTEMA AUTONOMO DE BAJO COSTE DE LA MONITORIZACION DE LA CALIDAD DEL AGUA EN TIEMPO REAL. MADRID, ESPAÑA.

F.MORENO. (2004). SISTEMA DE CONTROL SUPERVISOR DE LAS CONDICIONES AMBIENTALES DE UNA BODEGA DE PECES ORNAMENTALES. BOGOTA, COLOMBIA: TESIS.

GEEETECH.COM. (2014). Obtenido de https://www.geeetech.com/wiki/index.php/Arduino_GPRS_Shield

http://bibing.us.es/. (s.f.). *http://bibing.us.es/*. Obtenido de http://bibing.us.es/proyectos/abreproy/40048/fichero/VOLUMEN+1.+MEMORIA%252F4.+Tecnolog%C3%ADa+Bluetooth.pdf

http://www.appinventor.org/. (s.f.). *appinventor.org*. Obtenido de http://appinventor.mit.edu/explore/sites/all/files/hourofcode/AppInventorTutorials.pdf

http://www.electrobiomedical.com.co/. (s.f.). *http://www.electrobiomedical.com.co/*. Obtenido de http://www.electrobiomedical.com.co/download/datasheet/SEN0013.pdf

http://www.micropik.com/. (s.f.). *http://www.micropik.com*. Obtenido de http://www.micropik.com/PDF/HCSR04.pdf

http://www.uca.es/. (s.f.). *http://www.uca.es/*. Obtenido de http://www.uca.es/recursos/doc/Unidades/Unidad_Innovacion/Innovacion_Docente/ANEXOS_2011_2012/22232441_310201212102.pdf

https://www.olimex.com. (s.f.). *https://www.olimex.com*. Obtenido de https://www.olimex.com/Products/Components/RF/BLUETOOTH-SERIAL-HC-06/resources/hc06.pdf

I+D ELECTRONICA. (2016). Obtenido de I+D ELECTRONICA: http://www.didacticaselectronicas.com/index.php/sensores/sensor-de-gas-mq1352016-02-08-04-34-49_-detail

learn.mikroe.com. (2009). Obtenido de https://learn.mikroe.com/ebooks/microcontroladorespicc/chapter/lenguajes-de-programacion/

LEWIS, F. L. (2004). *Technologies, Protocols, and Applications*. New York: D.J. Cook and S.K. Das, John Wiley.

N. Aakvaag, J. E. (2006). Redes de sensores Inalambricos. *ABB*.

nv50.0fees.net. (s.f.). Obtenido de http://nv50.0fees.net/wp-content/uploads/manualproteus.pdf?ckattempt=1.(11)

Omar E. Barra Zapata, F. B. (2011). *Microcontroladores PIC con Programacion PBP*. MADRID: RAMA.

Pascual, F. R. (s.f.). Redes de sensores inalambricas. En F. R. Pascual, *Redes de sensores inalambricas*. Universidad Politecnica de Valencia.

R. F. Martínez, J. O. (2009). Redes Inalambricas de Sensores. En J. O. R. F. Martínez, *Redes Inalambricas de Sensores*. Universidad de Rioja.

RHYDOLABZ.COM. (2011). Obtenido de http://www.rhydolabz.com/documents/gps_gsm/sim900_rs232_gsm_modem_opn.pdf.(7)

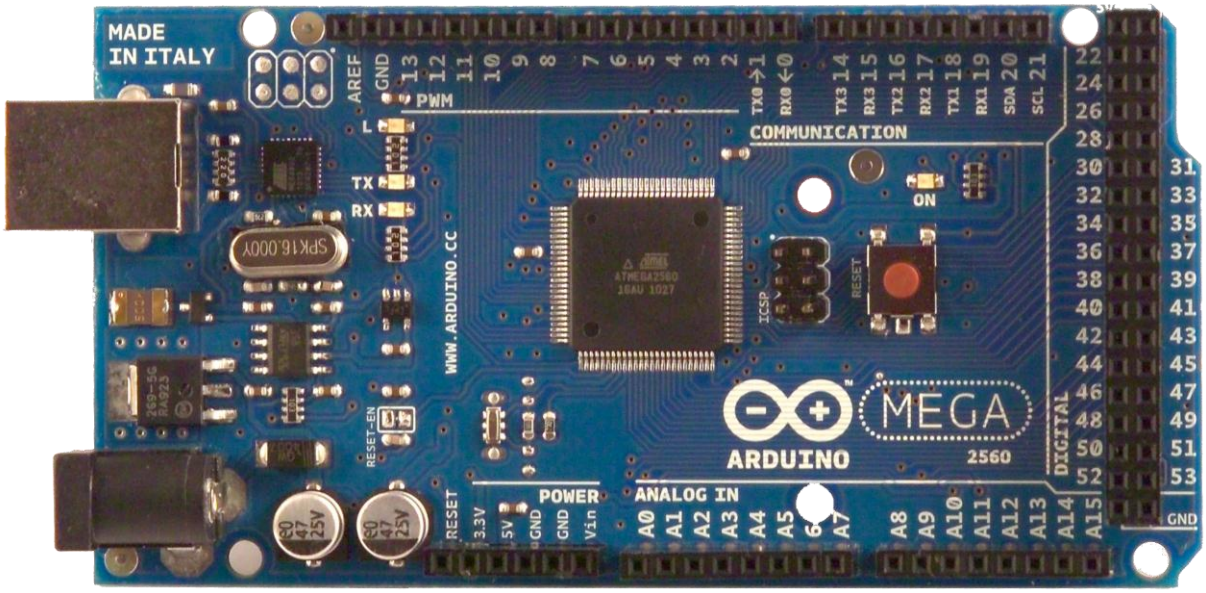
RODRIGUEZ, A. L. (2017). DISEÑO DE SISTEMA ELECTRONICO PARA DETECCION DE PRESENCIAS EXTERNAS EN VEHICULOS PESADOS. Piura, Piura, Peru.

saber.patagoniatec.com. (s.f.). Obtenido de http://saber.patagoniatec.com/usb-ttl-puerto-de-comunicacion-uart-arduino-argentina-ptec/

Yeferson Bedoya Giraldo, C. S. (2013). *Implementación, control y monitoreo de un Sistema de seguridad vehivular por redes gsm/gprs.* Universidad Tecnologica de Pereira, Pereira.

ANEXOS

Arduino MEGA 2560



Product Overview

The Arduino Mega 2560 is a microcontroller board based on the ATmega2560 (datasheet). It has 54 digital input/output pins (of which 14 can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The Mega is compatible with most shields designed for the Arduino Duemilanove or Diecimila.

Index

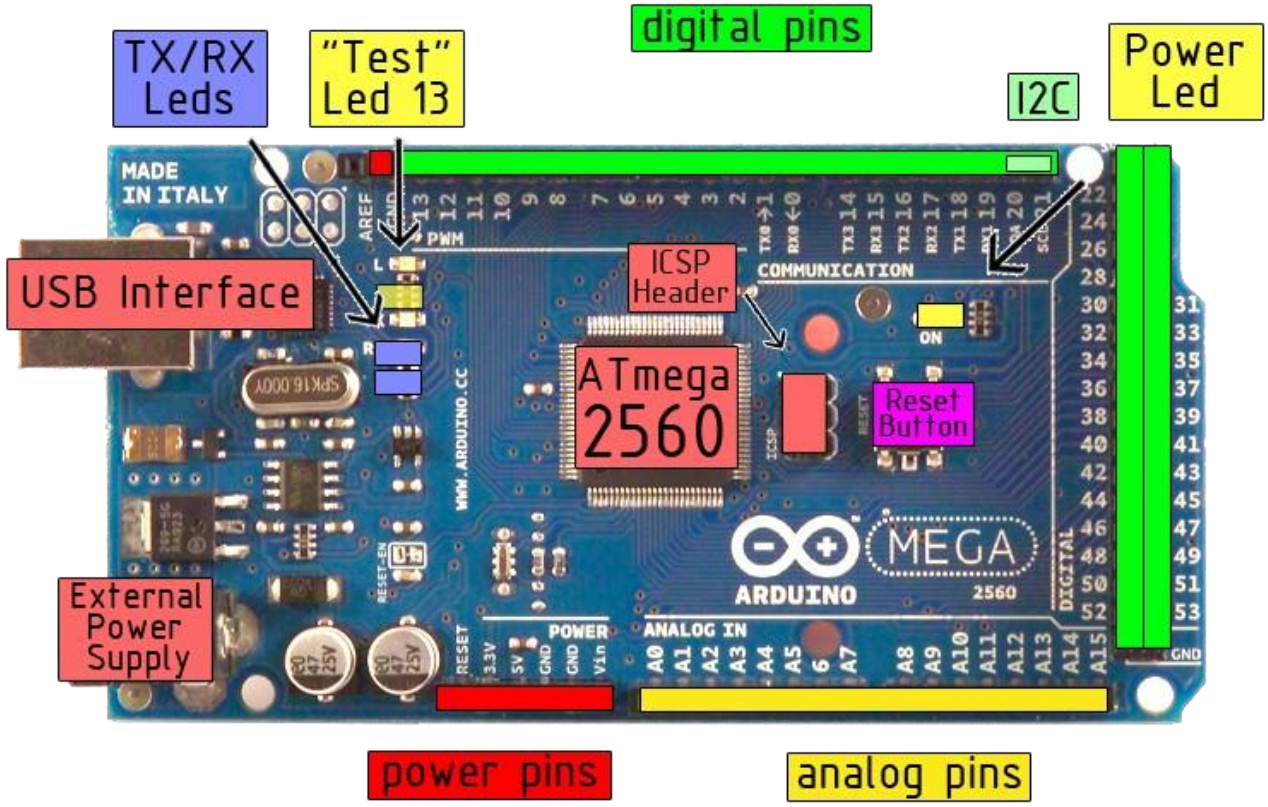
Technical Specification

EAGLE files: arduino-mega2560-reference-design.zip Schematic: arduino-mega2560-schematic.pdf

Summary

Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	54 (of which 14 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz

the board



Power

The Arduino Mega2560 can be powered via the USB connection or with an external power supply. The power source is selected automatically. External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector.

The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

The Mega2560 differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega8U2 programmed as a USB-to-serial converter.

The power pins are as follows:

VIN. The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.

5V. The regulated power supply used to power the microcontroller and other components on the board. This can come either from VIN via an on-board regulator, or be supplied by USB or another regulated 5V supply.

3V3. A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.

GND. Ground pins.

Memory

The ATmega2560 has 256 KB of flash memory for storing code (of which 8 KB is used for the bootloader), 8 KB of SRAM and 4 KB of EEPROM (which can be read and written with the EEPROM library).

Input and Output

Each of the 54 digital pins on the Mega can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions:

Serial: 0 (RX) and 1 (TX); Serial 1: 19 (RX) and 18 (TX); Serial 2: 17 (RX) and 16 (TX); Serial 3: 15 (RX) and 14 (TX). Used to receive (RX) and transmit (TX) TTL serial data. Pins 0 and 1 are also connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip .

External Interrupts: 2 (interrupt 0), 3 (interrupt 1), 18 (interrupt 5), 19 (interrupt 4), 20 (interrupt 3), and 21 (interrupt 2). These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.

PWM: 0 to 13. Provide 8-bit PWM output with the analogWrite() function.

SPI: 50 (MISO), 51 (MOSI), 52 (SCK), 53 (SS). These pins support SPI communication, which, although provided by the underlying hardware, is not currently included in the Arduino language. The SPI pins are also broken out on the ICSP header, which is physically compatible with the Duemilanove and Diecimila.

LED: 13. There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.

I^2C: 20 (SDA) and 21 (SCL). Support I^2C (TWI) communication using the Wire library (documentation on the Wiring website). Note that these pins are not in the same location as the I^2C pins on the Duemilanove.

The Mega2560 has 16 analog inputs, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and analogReference() function.

There are a couple of other pins on the board:

AREF. Reference voltage for the analog inputs. Used with analogReference().

Reset. Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

Communication

The Arduino Mega2560 has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega2560 provides four hardware UARTs for TTL (5V) serial communication. An ATmega8U2 on the board channels one of these over USB and provides a virtual com port to software on the computer (Windows machines will need a .inf file, but OSX and Linux machines will recognize the board as a COM port automatically. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the ATmega8U2 chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A SoftwareSerial library allows for serial communication on any of the Mega's digital pins.

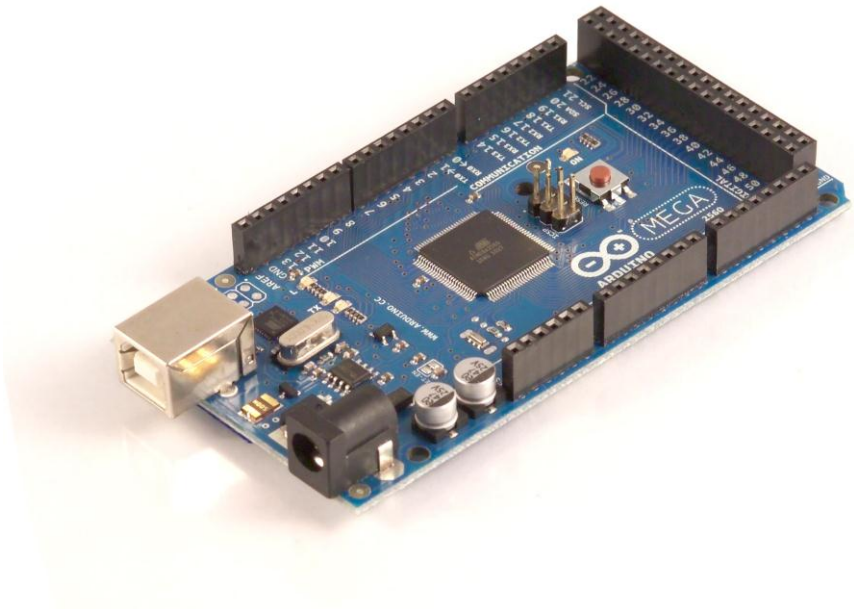
The ATmega2560 also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentation on the Wiring website for details. To use the SPI communication, please see the ATmega2560 datasheet.

Programming

The Arduino Mega2560 can be programmed with the Arduino software (download). For details, see the reference and tutorials.

The Atmega2560 on the Arduino Mega comes preburned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol (reference, C header files).

You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header; see these instructions for details.



Automatic (Software) Reset

Rather then requiring a physical press of the reset button before an upload, the Arduino Mega2560 is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2 is connected to the reset line of the ATmega2560 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment. This means that the bootloader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload.

This setup has other implications. When the Mega2560 is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Mega2560. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data.

The Mega contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it. It's labeled "RESET-EN". You may also be able to disable the auto-reset by connecting a 110 ohm resistor from 5V to the reset line; see this forum thread for details.

USB Overcurrent Protection

The Arduino Mega has a resettable polyfuse that protects your computer's USB ports from shorts and overcurrent. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

Physical Characteristics and Shield Compatibility

The maximum length and width of the Mega PCB are 4 and 2.1 inches respectively, with the USB connector and power jack extending beyond the former dimension. Three screw holes allow the board to be attached to a surface or case. Note that the distance between digital pins 7 and 8 is 160 mil (0.16"), not an even multiple of the 100 mil spacing of the other pins.

The Mega is designed to be compatible with most shields designed for the Diecimila or Duemilanove. Digital pins 0 to 13 (and the adjacent AREF and GND pins), analog inputs 0 to 5, the power header, and ICSP header are all in equivalent locations. Further the main UART (serial port) is located on the same pins (0 and 1), as are external interrupts 0 and 1 (pins 2 and 3 respectively). SPI is available through the ICSP header on both the Mega and Duemilanove / Diecimila. **Please note that I^2C is not located on the same pins on the Mega (20 and 21) as the Duemilanove / Diecimila (analog inputs 4 and 5).**

How to use Arduino

Arduino can sense the environment by receiving input from a variety of sensors and can affect its surroundings by controlling lights, motors, and other actuators. The microcontroller on the board is programmed using the Arduino programming language (based on Wiring) and the Arduino development environment (based on Processing). Arduino projects can be stand-alone or they can communicate with software on running on a computer (e.g. Flash, Processing, MaxMSP).

Arduino is a cross-platoform program. You'll have to follow different instructions for your personal OS. Check on the Arduino site for the latest instructions. *http://arduino.cc/en/Guide/HomePage*

Linux Install

Windows Install

Mac Install

Once you have downloaded/unzipped the arduino IDE, you can Plug the Arduino to your PC via USB cable.

Blink led

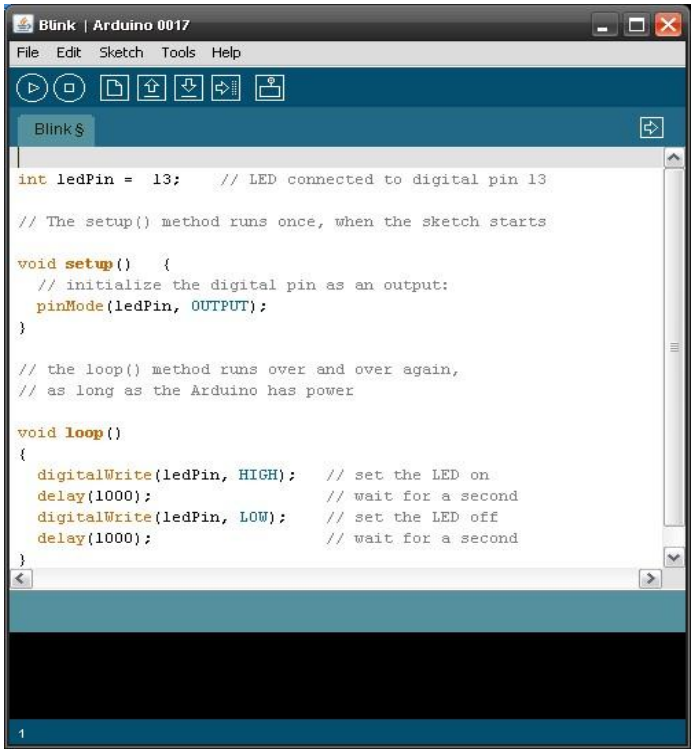
Now you're actually ready to "burn" your first program on the arduino board. To select "blink led", the physical translation of the well known programming "hello world", select

File>Sketchbook>
Arduino-0017>Examples>
Digital>Blink

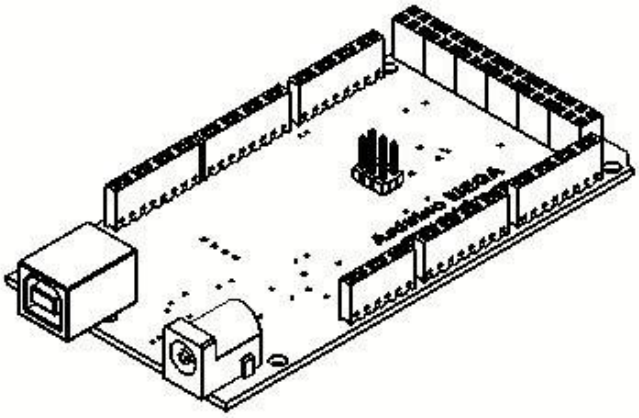
Once you have your skecth you'll see something very close to the screenshot on the right.

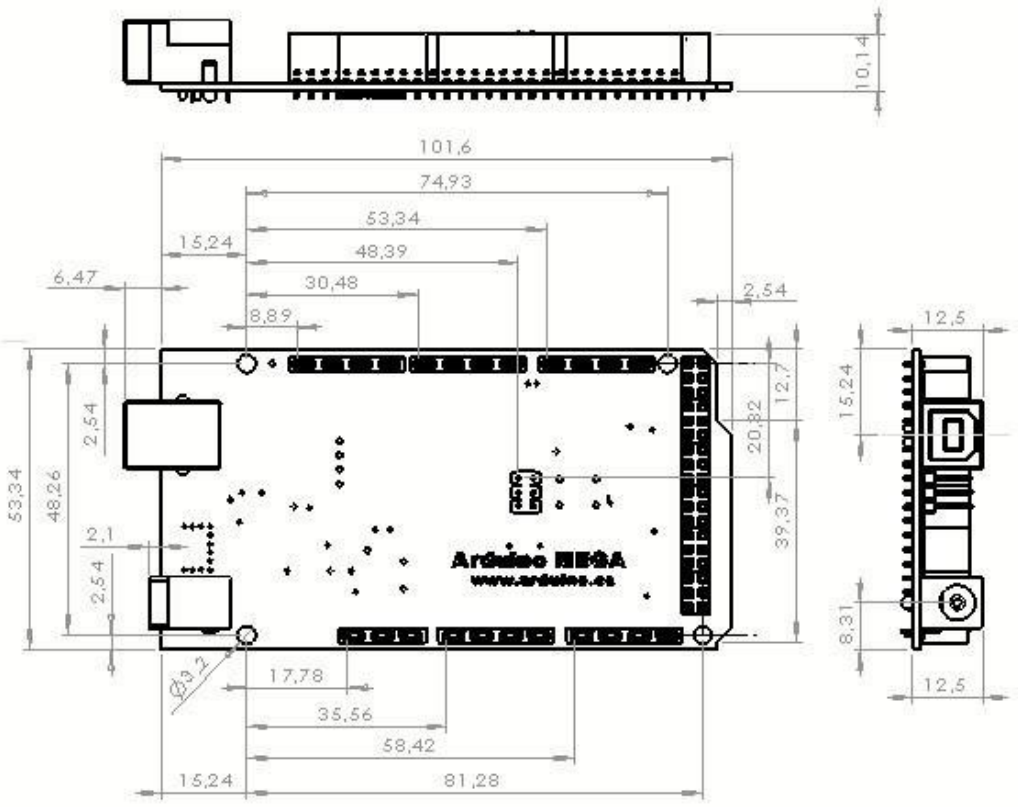
In **Tools>Board** select MEGA

Now you have to go to
Tools>SerialPort
and select the right serial port, the one arduino is attached to.



Dimensioned Drawing





Terms & Conditions

1. Warranties

1.1 The producer warrants that its products will conform to the Specifications. This warranty lasts for one (1) years from the date of the sale. The producer shall not be liable for any defects that are caused by neglect, misuse or mistreatment by the Customer, including improper installation or testing, or for any products that have been altered or modified in any way by a Customer. Moreover, The producer shall not be liable for any defects that result from Customer's design, specifications or instructions for such products. Testing and other quality control techniques are used to the extent the producer deems necessary.

1.2 If any products fail to conform to the warranty set forth above, the producer's sole liability shall be to replace such products. The producer's liability shall be limited to products that are determined by the producer not to conform to such warranty. If the producer elects to replace such products, the producer shall have a reasonable time to replacements. Replaced products shall be warranted for a new full warranty period.

1.3 EXCEPT AS SET FORTH ABOVE, PRODUCTS ARE PROVIDED "AS IS" AND "WITH ALL FAULTS." THE PRODUCER DISCLAIMS ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, REGARDING PRODUCTS, INCLUDING BUT NOT LIMITED TO, ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE

1.4 Customer agrees that prior to using any systems that include the producer products, Customer will test such systems and the functionality of the products as used in such systems. The producer may provide technical, applications or design advice, quality characterization, reliability data or other services. Customer acknowledges and agrees that providing these services shall not expand or otherwise alter the producer's warranties, as set forth above, and no additional obligations or liabilities shall arise from the producer providing such services.

1.5 The Arduino products are not authorized for use in safety-critical applications where a failure of the product would reasonably be expected to cause severe personal injury or death. Safety-Critical Applications include, without limitation, life support devices and systems, equipment or systems for the operation of nuclear facilities and weapons systems. Arduino products are neither designed nor intended for use in military or aerospace applications or environments and for automotive applications or environment. Customer acknowledges and agrees that any such use of Arduino products which is solely at the Customer's risk, and that Customer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

1.6 Customer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products and any use of Arduino products in Customer's applications, notwithstanding any applications-related information or support that may be provided by the producer.

2. Indemnification

The Customer acknowledges and agrees to defend, indemnify and hold harmless the producer from and against any and all third-party losses, damages, liabilities and expenses it incurs to the extent directly caused by: (i) an actual breach by a Customer of the representation and warranties made under this terms and conditions or (ii) the gross negligence or willful misconduct by the Customer.

3. Consequential Damages Waiver

In no event the producer shall be liable to the Customer or any third parties for any special, collateral, indirect, punitive, incidental, consequential or exemplary damages in connection with or arising out of the products provided hereunder, regardless of whether the producer has been advised of the possibility of such damages. This section will survive the termination of the warranty period.

4. Changes to specifications

The producer may make changes to specifications and product descriptions at any time, without notice. The Customer must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." The producer reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The product information on the Web Site or Materials is subject to change without notice. Do not finalize a design with this information.

Enviromental Policies

The producer of Arduino has joined the Impatto Zero® policy of LifeGate.it. For each Arduino board produced is created / looked after half squared Km of Costa Rica's forest's.

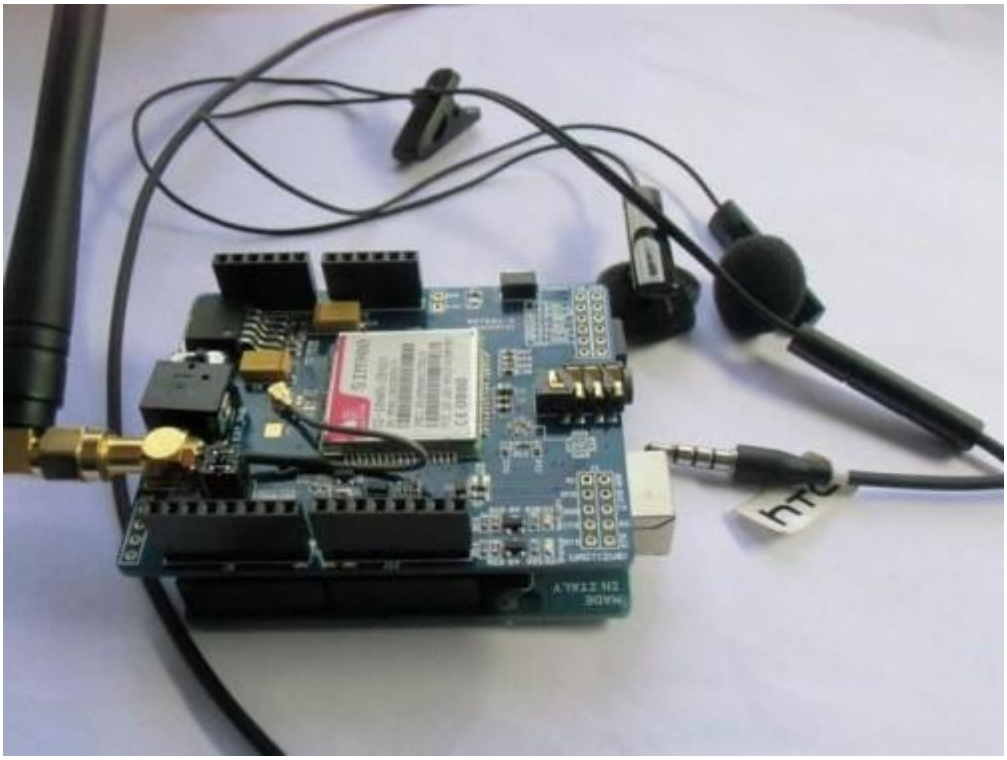
AZL GPRS/GSM SHIELD SIM900

Introduction

The GPRS/GSM Shield provides you a way to use the GSM cell phone network to receive data from a remote location. The shield allows you to achieve this via any of the three methods:

- Short Message Service
- Audio
- GPRS Service

The GPRS/GSM Shield is compatible with all boards which have the same form factor (and pin out) as a standard Arduino Board. The GPRS/GSM Shield is configured and controlled via its UART using simple AT commands. Based on the SIM900 module from SIMCOM, it is like a cell phone. Besides the communications features, the GPRS/GSM Shield has 6 GPIOs, 2 PWMs and an ADC.



Features

- **Quad-Band 850 / 900/ 1800 / 1900 MHz** - would work on GSM networks in all countries across the world.
- **GPRS multi-slot class 10/8**
- **GPRS mobile station class B**
- **Compliant to GSM phase 2/2+**
  - **Class 4 (2 W @ 850 / 900 MHz)**
  - **Class 1 (1 W @ 1800 / 1900MHz)**
- **Control via AT commands** - Standard Commands: GSM 07.07 & 07.05 | Enhanced Commands: SIMCOM AT Commands.

- **Short Message Service** - so that you can **send and receive** small amounts of data over the network (ASCII or raw hexadecimal).
- **Embedded TCP/UDP stack** - allows you to upload data to a web server.
- **RTC supported.**
- **Selectable serial port.**
- **2 in 1 head set jack**
- **Low power consumption** - 1.5mA(sleep mode)
- **Industrial Temperature Range** - -40°C to +85°C

Application Ideas

- M2M (Machine 2 Machine) Applications - To transfer control data using SMS or GPRS between two machines located at two different factories.
- Remote control of appliances - Send SMS while you are at your office to turn on or off your washing machine at home.
- Remote Weather station or a Wireless Sensor Network - Mate it with [AZLduino v1.0] and create a sensor node capable of transferring sensor data (like from a weather station - temperature, humidity etc.) to a web server (like pachube.com).
- Vehicle Tracking System - Couple the GPRS Shield with an Arduino and GPS module and install it in your car and publish your location live on the internet. Can be used as a automotive burglar alarm.

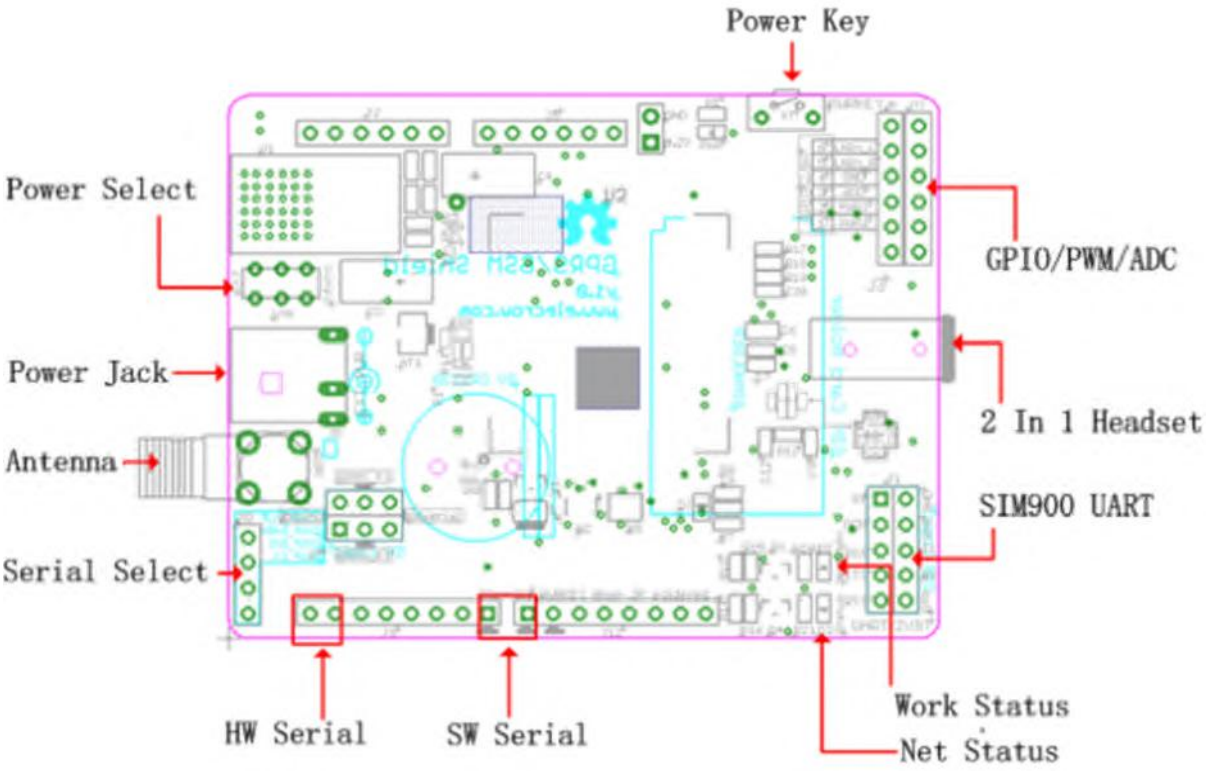
Cautions

- Make sure your SIM card is unlocked.
- The product is provided as is without an insulating enclosure. Please observe ESD precautions especially in dry (low humidity) weather.
- The factory default setting for the GPRS Shield UART is 19200 bps 8-N-1. (Can be changed using AT commands).

Specifications

Item	Min	Typical	Max	Unit
Voltage	4.8	5.0	5.2	VDC
Current	/	50	450	mA
Dimension(with antenna)		110x58x19		mm
Net Weight		47±2		g

Interface Function



Power select - select the power supply for GPRS shield(external power or 5v of Arduino)

Power jack - connected to external 4.8~5VDC power supply

Antenna interface - connected to external antenna

Serial port select - select either software serial port or hardware serial port to be connected to GPRS Shield

Hardware Serial - D0/D1 of Arduino//Seeeduino

Software serial - D7/D8 of Arduino/ Seeeduino only

Status LED - tell whether the power of SIM900 is on

Net light - tell the status about SIM900 linking to the net

UART of SIM900 - UART pins breakout of SIM900

Microphone - to answer the phone call

Speaker - to answer the phone call

GPIO,PWM and ADC of SIM900 - GPIO,PWM and ADC pins breakout of SIM900

Power key - power up and down for SIM900

Pins usage on Arduino

D0 - Unused if you select hardware serial port to communicate with GPRS Shield

D1 - Unused if you select hardware serial port to communicate with GPRS Shield

D2 - Unused

D3 - Unused

D4 - Unused

D5 - Unused

D6 - Unused

D7 - Used if you select software serial port to communicate with GPRS Shield

D8 - Used if you select software serial port to communicate with GPRS Shield

D9 - Used for software control the power up or down of the SIM900

D10 - Unused

D11 - Unused

D12 - Unused

D13 - Unused

D14(A0) - Unused

D15(A1) - Unused

D16(A2) - Unused

D17(A3) - Unused

D18(A4) - Unused

D19(A5) - Unused

Light Status

LED	Status	Function
Power-on indicator(Green)	Off	Power of GPRS Shield is off
	On	Power of GPRS Shield is on
Status Indicator(Red)	Off	Power off
	On	Power on
Net indicator(Green)	Off	SIM900 is not working
	64ms On/800ms Off	SIM900 does not find the network
	64ms On/3000ms Off	SIM900 finds the network
	64ms On/300ms Off	GPRS communication

Usage

Hardware installation

- **Insert an unlocked SIM card to SIM Card Holder** - 6 Pin Holder for SIM Cards. Both 1.8 volts and 3.0 volts SIM Cards are supported by SIM900 - the SIM card voltage type is automatically detected.


- **Make sure the antenna pad buckled properly** - A miniature coaxial RF connector is present on the GPRS Shield board to connect with a GSM Antenna. The connector present on the GPRS Shield is called a U.FL connecto. The GSM Antenna supplied with the GPRS Shield has an SMA connector (and not an RP-SMA connector) on it. A patch cord is

also supplied with the GPRS Shield to interface the antenna to the board. The connection topology is shown in the diagram below:

Antenna pad properly buckled

- **Assemble the GSM antenna**

Assemble the GSM antenna

- **Power supply for GPRS shield** - Select power source with the switch on board, you can select the 5V power supply from Arduino or external power. Select the 5V source from Arduino as the following picture:

select 5v of Arduino

- **Turn on the GPRS shield**--There is two ways to turn on the GPRS Shield.

1. Turn on through Hardware. Press the 'POWERKEY' for few seconds until Power-on indicator (Green) is on.

POWER ON key

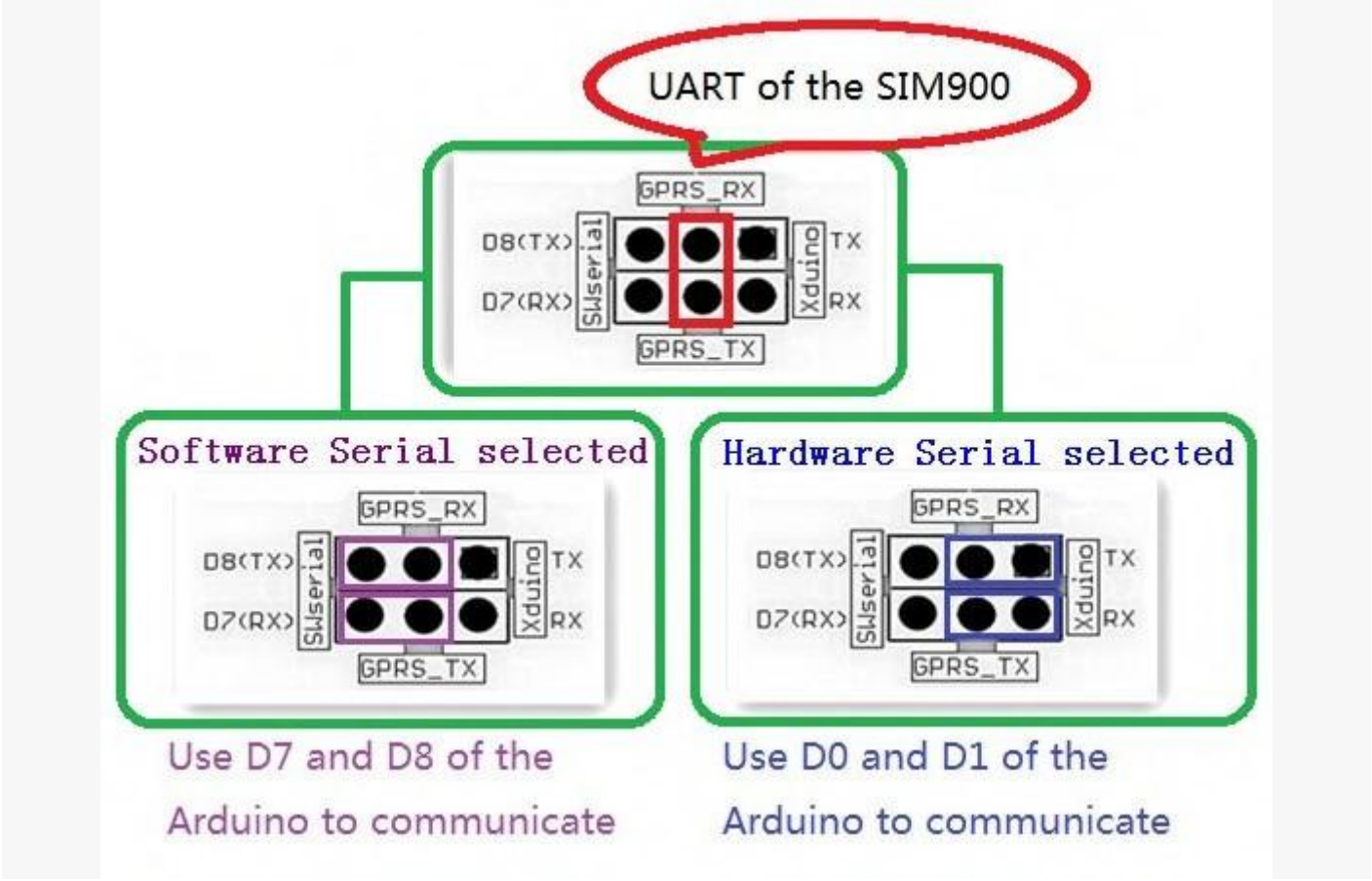
2. Turn on through Software. If the JP is soldered, run the following code, the GPRS will POWER on or POWER off.

```
void power_ON_Down()

pinMode(9, OUTPUT);
digitalWrite(9,LOW);
delay(1000);
digitalWrite(9,HIGH);
delay(2000);
digitalWrite(9,LOW);
delay(3000);
```

- **Serial Port(UART) Communication**

The GPRS Shield is used UART protocol to communicate with an Arduino/Arduino clone; Users can use jumpers to connect (RX,TX) of the shield to either Software Serial(D8,D7) or Hardware Serial(D1,D0) of the Arduino .Detailed information is showed as the following picture:



Selectable GPRS Shield Communication Port

Note:

- Users can use **"AT+IPR=?"** command to see supported baudrate, it will response a list of supported baudrate.
- Users can use **"AT+IPR=x"**("x" is value of supported baud rate) to set a fixed baud rate and save the configuration to non-volatile flash memory.
  - When users select Software Serial to communicate, Software Serial Library (zip) should be installed in Arduino's libraries.

- **Plug to Arduino UNO R3** - The GPRS Shield, like any other well designed shield, is stackable as shown in the photo below.

Power Down the GPRS Shield

The GPRS Shield can be turned off by following ways:

- **1, Normal power down procedure**: Turn off the GPRS shield by using **Hardware Trigger**; Press the **ON/OFF Button** about two seconds.

The power down scenarios illustrates as following figure:

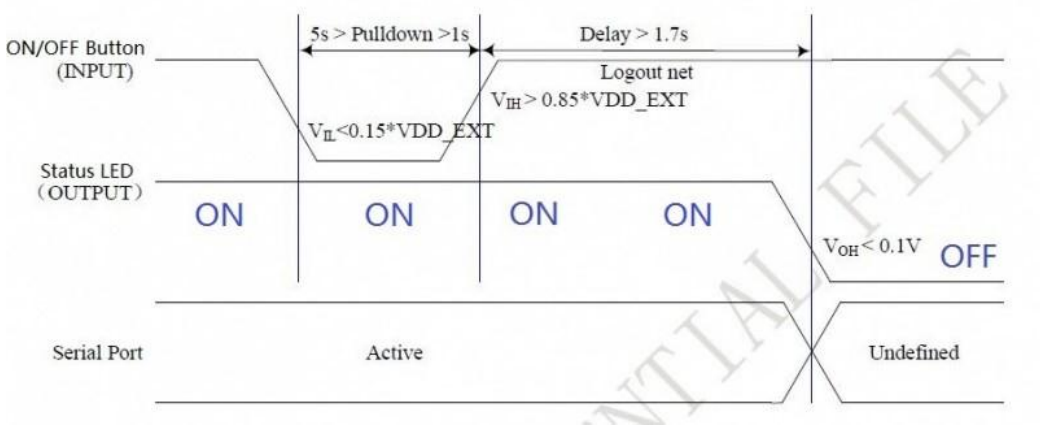


Figure of Timing of turning off GPRS Shield using Hardware Triger

- **2, Normal power down procedure**: If JP is soldered, then give Digital Pin 9 of the Arduino(act as Software Triger) a Turn off Impulse can turn off the GPRS Shield. The power down scenarios illustrates as following figure:

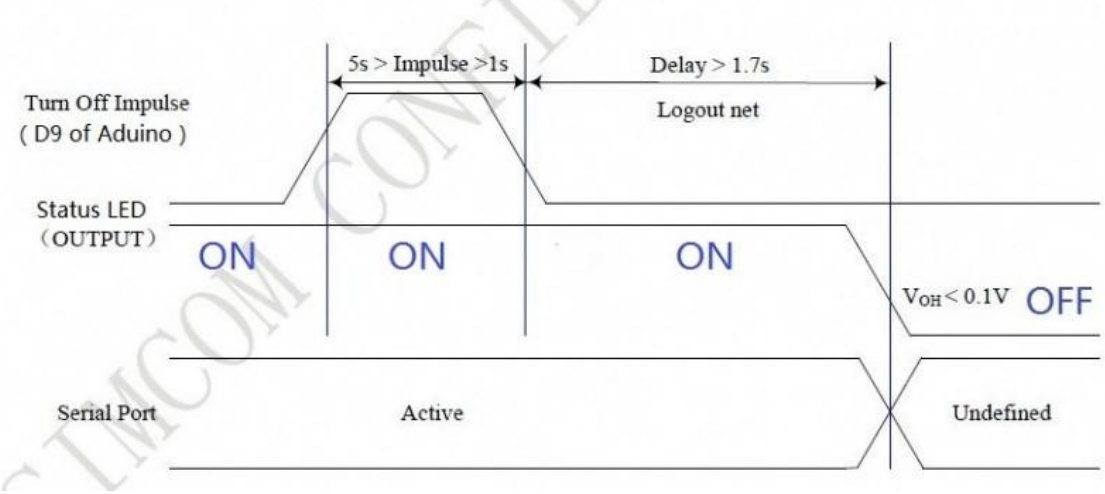


Figure of Timing of turning off GPRS Shield using Software Trigger

The following code is power down subroutine for Arduino if using software trigger:

```
void powerDown()
{
pinMode(9, OUTPUT);
digitalWrite(9,LOW);
delay(1000);
digitalWrite(9,HIGH);
delay(2000);
digitalWrite(9,LOW);
delay(3000);
```

```
}
```

- **3, Normal power down procedure**: Turn off the GPRS shield by sending AT command **"AT+CPOWD=1"** to SIM900 module.

When GPRS Shield power down in **Normal power down procedure**, the procedure lets the SIM900 log off from the network and allows the software to enter into a secure state and save data before completely disconnecting the power supply. Before the completion of the power down procedure the SIM900 will send out result code:

NORMAL POWER DOWN

- **4, Over-voltage or Under-voltage Automatic Power Down**: SIM900 will constantly monitor the voltage applied on the VBAT.

①If the voltage ≤ 3.3V, the following URC will be presented:

UNDER-VOLTAGE WARNNING

②If the voltage ≥ 4.7V, the following URC will be presented:

OVER-VOLTAGE WARNNING

③The uncritical voltage range is 3.2V to 4.8V. If the voltage > 4.8V or < 3.2V, SIM900 will be automatic power down soon. If the voltage < 3.2V, the following URC will be presented:

UNDER-VOLTAGE POWER DOWN

④If the voltage > 4.8V, the following URC will be presented:

OVER-VOLTAGE POWER DOWN

- **5, Over-temperature or Under-temperature Automatic Power Down**: SIM900 will constantly monitor the temperature of the module.

①If the temperature > 80℃, the following URC will be presented:

+CMTE:1

②If the temperature < -30℃, the following URC will be presented:

+CMTE:-1

③The uncritical temperature range is -40℃ to +85℃. If the temperature > +85℃ or < -40℃, the module will be automatic power down soon. If the temperature > +85℃, the following URC will be presented:

+CMTE:2

④If the temperature < -40℃, the following URC will be presented:

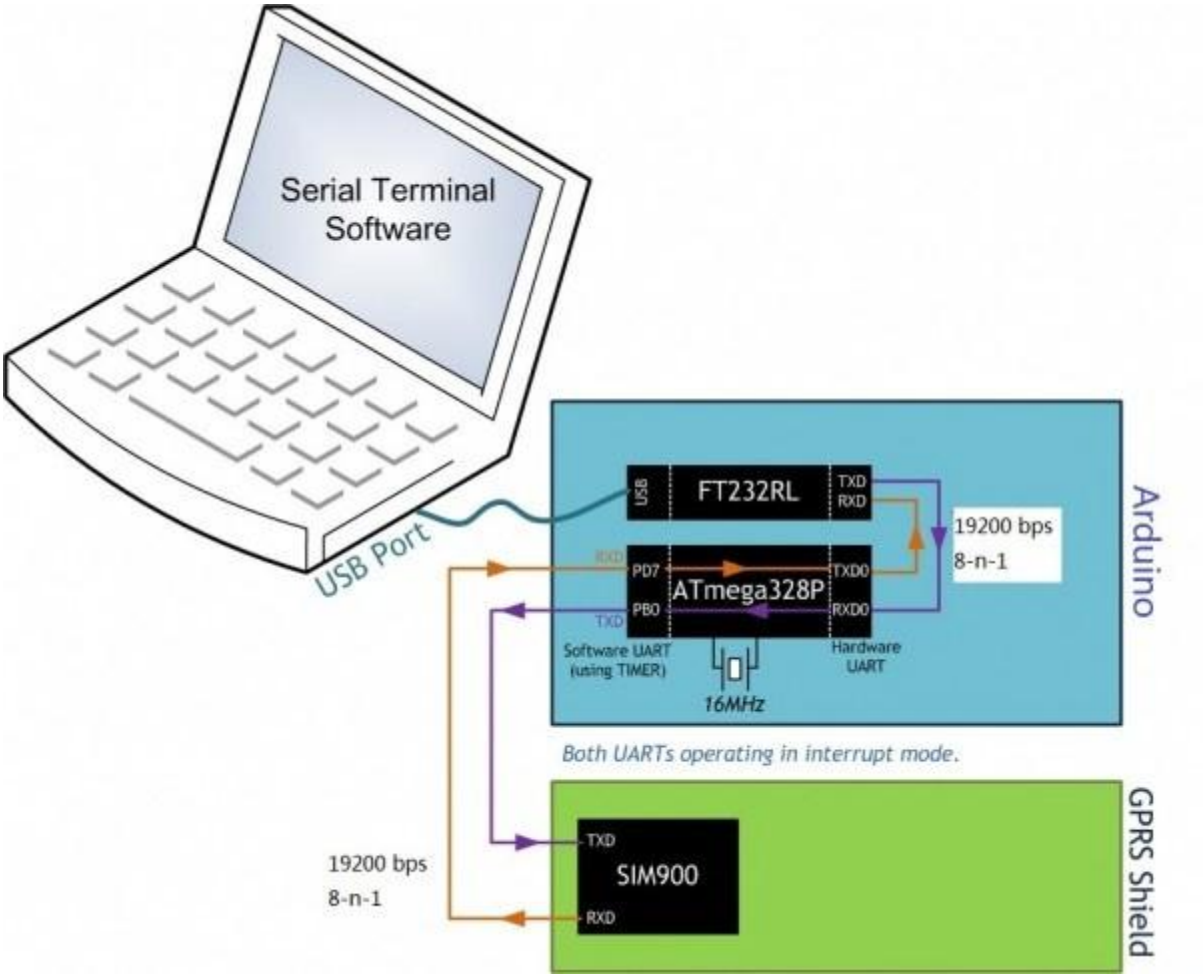
+CMTE:-2

When the GPRS Shield encounters POWER DOWN scenario, the AT commands cannot be executed. The SIM900 logs off from network and enters the POWER DOWN mode, only the RTC is still active. POWER DOWN can also be indicated by STATUS LED (Blue), which is off in this mode.

Note:

- To monitor the temperature, users can use the "**AT+CMTE**" command to read the temperature when GPRS Shield is powered on.
- To monitor the supply voltage, users can use the "**AT+CBC**" command which includes a parameter: voltage value (in mV) when GPRS Shield is powered on.

Upload Sketch to Arduino



Data Stream among Computer, Arduino and GPRS Shield

The following sketch configures Arduino/Arduino clone as serial link between PC and the GPRS Shield (Jumpers on SWserial side). PC would need a serial terminal software to communicate with it - Window's built-in HyperTerminal, Arduino IDE's Serial Monitor, Serial Terminals(sscom32) or Bray++ Terminal.

After uploading the sketch to the Arduino board, press the ON/OFF button on the GPRS Shield to turn it on; Now you can see what you get on the serial terminal and the status of the three indicator LEDs, then communicate with your Shield.

```
//Serial Relay - Arduino will patch a
//serial link between the computer and the GPRS Shield
//at 19200 bps 8-N-1
//Computer is connected to Hardware UART
//GPRS Shield is connected to the Software UART
```

```
#include <SoftwareSerial.h>

SoftwareSerial GSMSerial(7, 8);

void setup()
{
GSMSerial.begin(19200);               // the GPRS/GSM baud rate
Serial.begin(19200);                 // the GPRS/GSM baud rate
}

void loop()
{
if(Serial.available())

GSMSerial.print((char)Serial.read());

else  if(GSMSerial.available())

Serial.print((char)GSMSerial.read());
}
```

Note:

- The "AT" or "at" prefix must be set at the beginning of each Command line. To terminate a Command line enter <CR>.

Examples

Sending SMS: using Software UART

```
#include <SoftwareSerial.h>

SoftwareSerial mySerial(7, 8);

void setup()
{
mySerial.begin(19200);  //Default serial port setting for the GPRS modem is 19200bps 8-
N-1
mySerial.print("\r");
delay(1000);                    //Wait for a second while the modem sends an "OK"
mySerial.print("AT+CMGF=1\r");    //Because we want to send the SMS in text mode
delay(1000);

//mySerial.print("AT+CSCA=\"+919032055002\"\r");  //Setting for the SMS Message center
number,
//delay(1000);                                  //uncomment only if required and
replace with
//the message center number obtained from
```

```
//your GSM service provider.
//Note that when specifying a tring of characters
// " is entered as \"

mySerial.print("AT+CMGS=\"+9184460xxxx\"\r");    //Start accepting the text for the
message
//to be sent to the number specified.
//Replace this number with the target mobile number.
delay(1000);
mySerial.print("Hello,Elecrow!\r");   //The text for the message
delay(1000);
mySerial.write(0x1A);  //Equivalent to sending Ctrl+Z
}

void loop()
{
//We just want to send the SMS only once, so there is nothing in this loop.
//If we put the code for SMS here, it will be sent again and again and cost us a lot.
}
```

Making a call: using Software UART

```
#include <SoftwareSerial.h>

SoftwareSerial mySerial(7, 8);

void setup()
{
mySerial.begin(19200);               // the GPRS baud rate
Serial.begin(19200);               // the GPRS baud rate
delay(2000);
mySerial.println("ATDxxxxxxxxx;"); // xxxxxxxxx is the number you want to dial.

if(mySerial.available())

Serial.print((unsigned char)mySerial.read());



delay(10000);
delay(10000);

mySerial.println("ATH"); //End the call.
if(mySerial.available())

Serial.print((unsigned char)mySerial.read());
}
```

```
void loop()
{
//Do nothing
}
```

Using AT Commands to Control GPIO and PWM pins

Note: GPIOs,PWMs and ADC of the SIM900 module are all 2V8 logic.

```
#include <SoftwareSerial.h>

SoftwareSerial mySerial(7, 8);

void setup()
{
mySerial.begin(19200);               // the GPRS baud rate
Serial.begin(19200);               // the GPRS baud rate
delay(2000);
}

void loop()
{
mySerial.println("AT+SPWM=1,63,100");// set PWM 1 PIN
mySerial.println("AT+SPWM=2,63,50");// set PWM 2 PIN

mySerial.println("AT+SGPIO=0,1,1,1");// set GPIO 1 PIN to 1
mySerial.println("AT+SGPIO=0,12,1,1");
delay(1000);

mySerial.println("AT+SGPIO=0,1,1,0");// set GPIO 1 PIN to 0
mySerial.println("AT+SGPIO=0,12,1,0");
delay(1000);
}
```

A Simple Source Code Example

The demo code below is for the Xduino to send SMS message/dial a voice call/submit a http request to a website and upload data to the pachube. It has been tested on Arduino Duemilanove but will work on any compatible variant, please note that this sketch uses the software UART of ATmega328P. please follow the following steps for running this sketch.

1. With the GPRS Shield removed, download this sketch into your Arduino.
2. Disconnect the Xduino from USB port to remove power source.
3. Set the Serial Port jumpers on the GPRS Shield in SWserial position, to use the Soft Serial port of Arduino.
4. Connect the antenna to the GPRS Shield and insert the SIM Card.
5. Mount the GPRS Shield on Arduino.

6. Connect the Arduino to the computer by USB, and fire up your favorite serial terminal software on computer, choose the COM port for Arduino, set it to operate at 19200 8-N-1.
7. Type command in the terminal to execute different function, there are 4 functions in the demo:
    1. If you input 't', the demo will send a SMS message to another cellphone which you set(you need set the number in the code);
    2. If you input 'd', the program will dial a call to the other cellphone that you set(it is also need you set in the code);
    3. If you input 'h', it will submit a http request to a web that you want to access(it need you set the web address in the code), it will return a string from the website if it goes correctly;
    4. If you input 's', it will upload the data to the pachube (for detail you can refer to the explanation in the code). I strongly recommend you input 'h' before input 's', because uploading data to the pachube need do some setting, after execute the function of submit a http request, the setting will be set.
8. If the program returns error in the terminal after you typed the command, don't worry, just try input the command again.

```
/*Note: this code is a demo for how to using gprs shield to send sms message, dial a
voice call and
send a http request to the website, upload data to pachube.com by TCP connection,

The microcontrollers Digital Pin 7 and hence allow unhindered
communication with GPRS Shield using SoftSerial Library.
IDE: Arduino 1.0 or later
Replace the following items in the code:
1.Phone number, don't forget add the country code
2.Replace the Access Point Name
3. Replace the Pachube API Key with your personal ones assigned
to your account at cosm.com
*/


#include <SoftwareSerial.h>
#include <String.h>

SoftwareSerial mySerial(7, 8);

void setup()
{
mySerial.begin(19200);               // the GPRS baud rate
Serial.begin(19200);    // the GPRS baud rate
delay(500);
}

void loop()
{
```

```
//after start up the program, you can using terminal to connect the serial of gprs shield,
//if you input 't' in the terminal, the program will execute  GetSignalQuality(),it will show the signal quality,
//if you input 't' in the terminal, the program will execute SendTextMessage(), it will show how to send a sms message,
//if input 'd' in the terminal, it will execute DialVoiceCall(), etc.

if (Serial.available())
switch(Serial.read())

case 'q':
GetSignalQuality();
break;
case 't':
SendTextMessage();
break;
case 'd':
DialVoiceCall();
break;
case 'h':
SubmitHttpRequest();
break;
case 's':
Send2Pachube();
break;

if (mySerial.available())
Serial.write(mySerial.read());
}
///GetSignalQuality()
///get the signal quality of GSM model.
void GetSignalQuality()
{
mySerial.println("AT+CSQ");  //get the signal Quality
delay(100);
int k=0;
while(mySerial.available()!=0)

SigQ[k]=mySerial.read();
Serial.write(SigQ[k]);
k+=1;
}



///SendTextMessage()
///this function is to send a sms message
void SendTextMessage()
```

```
{
mySerial.print("AT+CMGF=1\r");    //Because we want to send the SMS in text mode
delay(100);
mySerial.println("AT + CMGS = \"+86138xxxxx615\"");//send sms message, be careful need
to add a country code before the cellphone number
delay(100);
mySerial.println("A test message!");//the content of the message
delay(100);
mySerial.println((char)26);//the ASCII code of the ctrl+z is 26
delay(100);
mySerial.println();
}

///DialVoiceCall
///this function is to dial a voice call
void DialVoiceCall()
{
mySerial.println("ATD + +86138xxxxx615;");//dial the number
delay(100);
mySerial.println();
}

///SubmitHttpRequest()
///this function is submit a http request
///attention:the time of delay is very important, it must be set enough
void SubmitHttpRequest()
{
mySerial.println("AT+CSQ");
delay(100);

ShowSerialData();// this code is to show the data from gprs shield, in order to easily
see the process of how the gprs shield submit a http request, and the following is for
this purpose too.

mySerial.println("AT+CGATT?");
delay(100);

ShowSerialData();

mySerial.println("AT+SAPBR=3,1,\"CONTYPE\",\"GPRS\"");//setting the SAPBR, the
connection type is using gprs
delay(1000);

ShowSerialData();

mySerial.println("AT+SAPBR=3,1,\"APN\",\"CMNET\"");//setting the APN, the second need
you fill in your local apn server
delay(4000);
```

```
ShowSerialData();

mySerial.println("AT+SAPBR=1,1");//setting the SAPBR, for detail you can refer to the
AT command mamual
delay(2000);

ShowSerialData();

mySerial.println("AT+HTTPINIT"); //init the HTTP request

delay(2000);
ShowSerialData();

mySerial.println("AT+HTTPPARA=\"URL\",\"www.google.com.hk\"");// setting the httppara,
the second parameter is the website you want to access
delay(1000);

ShowSerialData();

mySerial.println("AT+HTTPACTION=0");//submit the request
delay(10000);//the delay is very important, the delay time is base on the return from
the website, if the return datas are very large, the time required longer.
//while(!mySerial.available());

ShowSerialData();

mySerial.println("AT+HTTPREAD");// read the data from the website you access
delay(300);

ShowSerialData();

mySerial.println("");
delay(100);
}

///send2Pachube()///
///this function is to send the sensor data to the pachube, you can see the new value
in the pachube after execute this function///
void Send2Pachube()
{
mySerial.println("AT+CGATT?");
delay(100);

ShowSerialData();

mySerial.println("AT+CSTT=\"CMNET\"");//start task and setting the APN,
delay(1000);
```

```
ShowSerialData();

mySerial.println("AT+CIICR");//bring up wireless connection
delay(300);

ShowSerialData();

mySerial.println("AT+CIFSR");//get local IP adress
delay(2000);

ShowSerialData();

mySerial.println("AT+CIPSPRT=0");
delay(3000);

ShowSerialData();

mySerial.println("AT+CIPSTART=\"tcp\",\"api.cosm.com\",\"8081\"");//start up the
connection
delay(2000);

ShowSerialData();

mySerial.println("AT+CIPSEND");//begin send data to remote server
delay(4000);
ShowSerialData();
String humidity = "1031";//these 4 line code are imitate the real sensor data, because
the demo did't add other sensor, so using 4 string variable to replace.
String moisture = "1242";//you can replace these four variable to the real sensor data
in your project
String temperature = "30";//
String barometer = "60.56";//
mySerial.print("\"method\": \"put\",\"resource\": \"/feeds/43634/\",\"params\"");//here
is the feed you apply from pachube
delay(500);
ShowSerialData();
mySerial.print(": ,\"headers\": \"X-PachubeApiKey\":");//in here, you should replace
your pachubeapikey
delay(500);
ShowSerialData();
mySerial.print(" \"_cXwr5LE8qW4a296O-cDwOUvfddFer5pGmaRigPsiO0");//pachubeapikey
delay(500);
ShowSerialData();
mySerial.print("jEB9OjK-W6vej56j9ItaSlIac-hgbQjxExuveD95yc8BttXc");//pachubeapikey
delay(500);
ShowSerialData();
mySerial.print("Z7_seZqLVjeCOmNbEXUva45t6FL8AxOcuNSsQS\",\"body\":");
```

```
delay(500);
ShowSerialData();
mySerial.print(" \"version\": \"1.0.0\",\"datastreams\": ");
delay(500);
ShowSerialData();
mySerial.println("[\"id\": \"01\",\"current_value\": \"" + barometer + "\",");
delay(500);
ShowSerialData();
mySerial.println("\"id\": \"02\",\"current_value\": \"" + humidity + "\",");
delay(500);
ShowSerialData();
mySerial.println("\"id\": \"03\",\"current_value\": \"" + moisture + "\",");
delay(500);
ShowSerialData();
mySerial.println("\"id\": \"04\",\"current_value\": \"" + temperature + "\"],\"token\":
\"lee\"");


delay(500);
ShowSerialData();

mySerial.println((char)26);//sending
delay(5000);//waitting for reply, important! the time is base on the condition of
internet
mySerial.println();

ShowSerialData();

mySerial.println("AT+CIPCLOSE");//close the connection
delay(100);
ShowSerialData();

}
void ShowSerialData()
{
while(mySerial.available()!=0)
Serial.write(mySerial.read());
}
```

Using SMS to Control an LED Status

This example is contributed by MChobby, for more information please

visit: http://mchobby.be/wiki/index.php?title=SmsCommand

Send a SMS message "on" or "off" from your cellphone to the GPRS Shield to control the Digital Pin 13(LED) Status.

- The default Buffer of Rx in SoftwareSerial.h is 32/64, you may experience some data lose while the returns of SIM900 are many(Receiving SMS/TCPIP), you can try to change the Buffer of Rx in SoftwareSerial.h into

#define _SS_MAX_RX_BUFF *128* // RX buffer size

```
#include <SoftwareSerial.h>

SoftwareSerial mySerial(7, 8);

// EN: String buffer for the GPRS shield message

String msg = String("");
// EN: Set to 1 when the next GPRS shield message will contains the SMS message

int SmsContentFlag = 0;

// EN: Pin of the LED to turn ON and OFF depending on the received message

int ledPin = 13;

// EN: Code PIN of the SIM card (if applied)

//String SIM_PIN_CODE = String( "XXXX" );

void setup()
{
  mySerial.begin(19200);               // the GPRS baud rate
  Serial.begin(19200);                 // the GPRS baud rate

  // Initialize la PIN
  pinMode( ledPin, OUTPUT );
  digitalWrite( ledPin, LOW );
}

void loop()
{
    char SerialInByte;

    if(Serial.available())
    {
       mySerial.print((unsigned char)Serial.read());
     }
    else  if(mySerial.available())
    {
        char SerialInByte;
        SerialInByte = (unsigned char)mySerial.read();
```

```
        // EN: Relay to Arduino IDE Monitor

        Serial.print( SerialInByte );

        // ------------------------------------------------------------------
        // EN: Program also listen to the GPRS shield message.

        // ------------------------------------------------------------------

        // EN: If the message ends with <CR> then process the message

        if( SerialInByte == 13 ){
          // EN: Store the char into the message buffer

          ProcessGprsMsg();
         }
         if( SerialInByte == 10 ){
            // EN: Skip Line feed

         }
         else {
           // EN: store the current character in the message string buffer

           msg += String(SerialInByte);
         }
     }
}

// EN: Make action based on the content of the SMS.
//     Notice than SMS content is the result of the processing of several GPRS shield
messages.

void ProcessSms( String sms ){
  Serial.print( "ProcessSms for [" );
  Serial.print( sms );
  Serial.println( "]" );

  if( sms.indexOf("on") >= 0 ){
    digitalWrite( ledPin, HIGH );
    Serial.println( "LED IS ON" );
    return;
  }
  if( sms.indexOf("off") >= 0 ){
    digitalWrite( ledPin, LOW );
    Serial.println( "LED IS OFF" );
    return;
  }
```

```
}

// EN: Send the SIM PIN Code to the GPRS shield

//void GprsSendPinCode(){
//  if( SIM_PIN_CODE.indexOf("XXXX")>=0 ){
//    Serial.println( "*** OUPS! you did not have provided a PIN CODE for your SIM
CARD. ***" );
//    Serial.println( "*** Please, define the SIM_PIN_CODE variable . ***" );
//    return;
// }
//  mySerial.print("AT+CPIN=");
// mySerial.println( SIM_PIN_CODE );
}

// EN: Request Text Mode for SMS messaging

void GprsTextModeSMS(){
  mySerial.println( "AT+CMGF=1" );
}

void GprsReadSmsStore( String SmsStorePos ){
  // Serial.print( "GprsReadSmsStore for storePos " );
  // Serial.println( SmsStorePos );
  mySerial.print( "AT+CMGR=" );
  mySerial.println( SmsStorePos );
}

// EN: Clear the GPRS shield message buffer

void ClearGprsMsg(){
  msg = "";
}

// EN: interpret the GPRS shield message and act appropiately

void ProcessGprsMsg() {
  Serial.println("");
  Serial.print( "GPRS Message: [" );
  Serial.print( msg );
  Serial.println( "]" );

//  if( msg.indexOf( "+CPIN: SIM PIN" ) >= 0 ){
//     Serial.println( "*** NEED FOR SIM PIN CODE ***" );
 //    Serial.println( "PIN CODE *** WILL BE SEND NOW" );
 //    GprsSendPinCode();
//  }
```

```
  if( msg.indexOf( "Call Ready" ) >= 0 ){
     Serial.println( "*** GPRS Shield registered on Mobile Network ***" );
     GprsTextModeSMS();
  }

  // EN: unsolicited message received when getting a SMS message
  // FR: Message non sollicité quand un SMS arrive
  if( msg.indexOf( "+CMTI" ) >= 0 ){
     Serial.println( "*** SMS Received ***" );
     // EN: Look for the coma in the full message (+CMTI: "SM",6)
     //     In the sample, the SMS is stored at position 6
     int iPos = msg.indexOf( "," );
     String SmsStorePos = msg.substring( iPos+1 );
     Serial.print( "SMS stored at " );
     Serial.println( SmsStorePos );

     // EN: Ask to read the SMS store
     GprsReadSmsStore( SmsStorePos );
  }

  // EN: SMS store readed through UART (result of GprsReadSmsStore request)
  if( msg.indexOf( "+CMGR:" ) >= 0 ){
    // EN: Next message will contains the BODY of SMS
    SmsContentFlag = 1;
    // EN: Following lines are essentiel to not clear the flag!
    ClearGprsMsg();
    return;
  }

  // EN: +CMGR message just before indicate that the following GRPS Shield message
  //     (this message) will contains the SMS body

  if( SmsContentFlag == 1 ){
    Serial.println( "*** SMS MESSAGE CONTENT ***" );
    Serial.println( msg );
    Serial.println( "*** END OF SMS MESSAGE ***" );
    ProcessSms( msg );
  }

  ClearGprsMsg();
  // EN: Always clear the flag

  SmsContentFlag = 0;
}
```

SoftwareSerial library Notes

With Arduino 1.0 you should be able to use the SoftwareSerial library included with the distribution (instead of NewSoftSerial). However, you must be aware that the buffer reserved for incoming messages are hardcoded to 64 bytes in the library header, "SoftwareSerial.h":

```
#define _SS_MAX_RX_BUFF 64 // RX buffer size
```

This means that if the GPRS module responds with more data than that, you are likely to loose it with a buffer overflow! For instance, reading out an SMS from the module with "AT+CMGR=xx" (xx is the message index), you might not even see the message part because the preceding header information (like telephone number and time) takes up a lot of space. The fix seems to be to manually change _SS_MAX_RX_BUFF to a higher value (but reasonable so you don't use all you precious memory!)

The Softwareserial library has the following limitations (taken from Arduino page) If using multiple software serial ports, only one can receive data at a time .http://arduino.cc/hu/Reference/SoftwareSerial This means that if you try to add another serial device ie grove serial LCD you may get communication errors unless you craft your code carefully.

DS18B20

Programmable Resolution 1-Wire Digital Thermometer

General Description

The DS18B20 digital thermometer provides 9-bit to 12-bit Celsius temperature measurements and has an alarm function with nonvolatile user-programmable upper and lower trigger points. The DS18B20 communicates over a 1-Wire bus that by definition requires only one data line (and ground) for communication with a central microprocessor. In addition, the DS18B20 can derive power directly from the data line (“parasite power”), eliminating the need for an external power supply.

Each DS18B20 has a unique 64-bit serial code, which allows multiple DS18B20s to function on the same 1-Wire bus. Thus, it is simple to use one microprocessor to control many DS18B20s distributed over a large area. Applications that can benefit from this feature include HVAC environmental controls, temperature monitoring systems inside buildings, equipment, or machinery, and process monitoring and control systems.

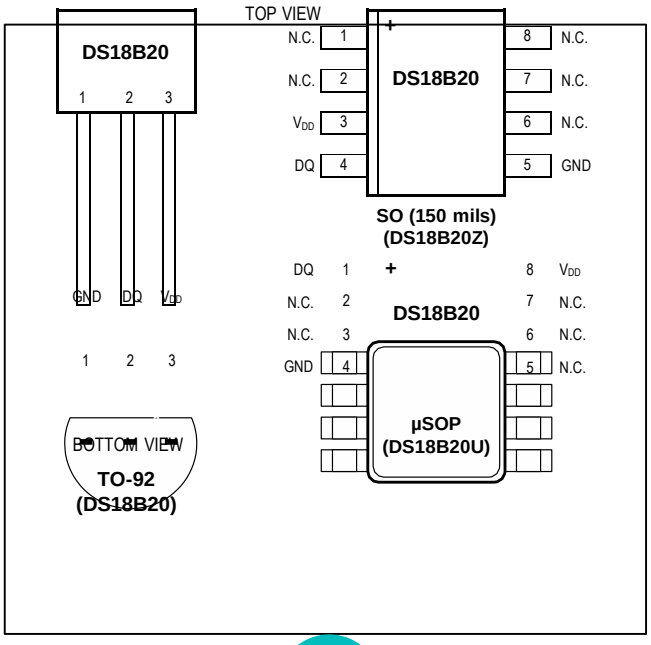
Applications

- Thermostatic Controls
- Industrial Systems
- Consumer Products
- Thermometers
- Thermally Sensitive Systems

Benefits and Features

- Unique 1-Wire® Interface Requires Only One Port Pin for Communication
- Reduce Component Count with Integrated Temperature Sensor and EEPROM
  - Measures Temperatures from -55°C to +125°C (-67°F to +257°F)
  - ±0.5°C Accuracy from -10°C to +85°C
  - Programmable Resolution from 9 Bits to 12 Bits
  - No External Components Required
- Parasitic Power Mode Requires Only 2 Pins for Operation (DQ and GND)
- Simplifies Distributed Temperature-Sensing Applications with Multidrop Capability
  - Each Device Has a Unique 64-Bit Serial Code Stored in On-Board ROM
- Flexible User-Definable Nonvolatile (NV) Alarm Settings with Alarm Search Command Identifies Devices with Temperatures Outside Programmed Limits
- Available in 8-Pin SO (150 mils), 8-Pin µSOP, and 3-Pin TO-92 Packages

Pin Configurations



Ordering Information **appears at end of data sheet.**

19-7487; Rev 4; 1/15

maxim integrated™

Absolute Maximum Ratings

Voltage Range on Any Pin Relative to Ground....-0.5V to +6.0V
Operating Temperature Range..... -55°C to +125°C
Storage Temperature Range..... -55°C to +125°C
Solder Temperature.....Refer to the IPC/JEDEC J-STD-020 Specification.

These are stress ratings only and functional operation of the device at these or any other conditions above those indicated in the operation sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods of time may affect reliability.

DC Electrical Characteristics

(-55°C to +125°C; V_{DD} = 3.0V to 5.5V)

PARAMETER	SYMBOL	CONDITIONS		MIN	TYP	MAX	UNITS
Supply Voltage	V_{DD}	Local power (Note 1)		+3.0		+5.5	V
Pullup Supply Voltage	V_{PU}	Parasite power	(Notes 1, 2)	+3.0		+5.5	V
		Local power		+3.0		V_{DD}	
Thermometer Error	t_{ERR}	-10°C to +85°C	(Note 3)			±0.5	°C
		-55°C to +125°C				±2	
Input Logic-Low	V_{IL}	(Notes 1, 4, 5)		-0.3		+0.8	V
Input Logic-High	V_{IH}	Local power	(Notes 1,6)	+2.2		The lower of 5.5 or V_{DD} + 0.3	V
		Parasite power		+3.0			
Sink Current	I_L	$V_{I/O}$ = 0.4V		4.0			mA
Standby Current	I_{DDS}	(Notes 7, 8)			750	1000	nA
Active Current	I_{DD}	V_{DD} = 5V (Note 9)			1	1.5	mA
DQ Input Current	I_{DQ}	(Note 10)			5		µA
Drift		(Note 11)			±0.2		°C

Note 1: All voltages are referenced to ground.
Note 2: The Pullup Supply Voltage specification assumes that the pullup device is ideal, and therefore the high level of the pullup is equal to V_{PU}. In order to meet the V_{IH} spec of the DS18B20, the actual supply rail for the strong pullup transistor must include margin for the voltage drop across the transistor when it is turned on; thus: $V_{PU\_ACTUAL} = V_{PU\_IDEAL} + V_{TRANSISTOR}$.
Note 3: See typical performance curve in Figure 1.
Note 4: Logic-low voltages are specified at a sink current of 4mA.
Note 5: To guarantee a presence pulse under low voltage parasite power conditions, V_{ILMAX} may have to be reduced to as low as 0.5V.
Note 6: Logic-high voltages are specified at a source current of 1mA.
Note 7: Standby current specified up to +70°C. Standby current typically is 3µA at +125°C.
Note 8: To minimize I_{DDS}, DQ should be within the following ranges: GND ≤ DQ ≤ GND + 0.3V or V_{DD} – 0.3V ≤ DQ ≤ V_{DD}.
Note 9: Active current refers to supply current during active temperature conversions or EEPROM writes.
Note 10: DQ line is high (“high-Z” state).
Note 11: Drift data is based on a 1000-hour stress test at +125°C with V_{DD} = 5.5V.

AC Electrical Characteristics–NV Memory

(-55°C to +125°C; V_{DD} = 3.0V to 5.5V)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
NV Write Cycle Time	t_{WR}			2	10	ms
EEPROM Writes	N_{EEWR}	-55°C to +55°C	50k			writes
EEPROM Data Retention	t_{EEDR}	-55°C to +55°C	10			years

AC Electrical Characteristics

(-55°C to +125°C; V_{DD} = 3.0V to 5.5V)

PARAMETER	SYMBOL	CONDITIONS		MIN	TYP	MAX	UNITS
Temperature Conversion Time	t_{CONV}	9-bit resolution	(Note 12)			93.75	ms
		10-bit resolution				187.5	
		11-bit resolution				375	
		12-bit resolution				750	
Time to Strong Pullup On	t_{SPON}	Start convert T command issued				10	µs
Time Slot	t_{SLOT}	(Note 12)		60		120	µs
Recovery Time	t_{REC}	(Note 12)		1			µs
Write 0 Low Time	t_{LOW0}	(Note 12)		60		120	µs
Write 1 Low Time	t_{LOW1}	(Note 12)		1		15	µs
Read Data Valid	t_{RDV}	(Note 12)				15	µs
Reset Time High	t_{RSTH}	(Note 12)		480			µs
Reset Time Low	t_{RSTL}	(Notes 12, 13)		480			µs
Presence-Detect High	t_{PDHIGH}	(Note 12)		15		60	µs
Presence-Detect Low	t_{PDLOW}	(Note 12)		60		240	µs
Capacitance	$C_{IN/OUT}$					25	pF

Note 12: See the timing diagrams in Figure 2.
Note 13: Under parasite power, if t_{RSTL} > 960µs, a power-on reset can occur.

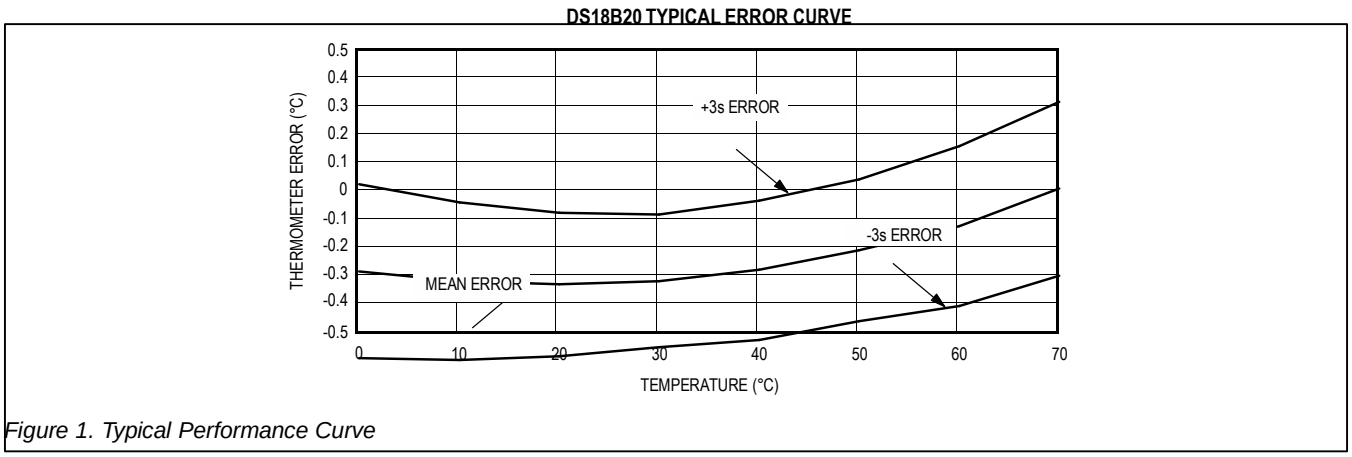


Figure 1. Typical Performance Curve

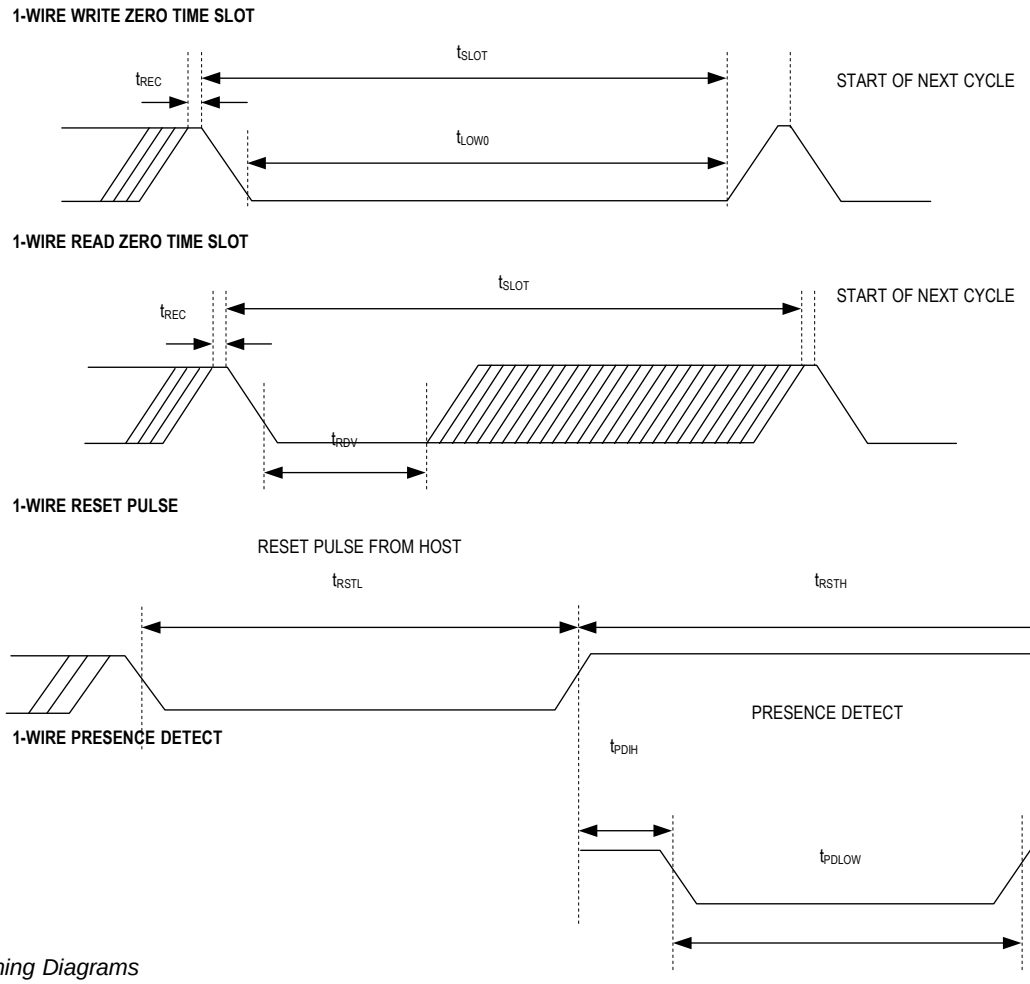


Figure 2. Timing Diagrams

Pin Description

PIN			NAME	FUNCTION
SO	µSOP	TO-92		
1, 2, 6, 7, 8	2, 3, 5, 6, 7	—	N.C.	No Connection
3	8	3	V_{DD}	Optional V_{DD}. V_{DD} must be grounded for operation in parasite power mode.
4	1	2	DQ	Data Input/Output. Open-drain 1-Wire interface pin. Also provides power to the device when used in parasite power mode (see the *Powering the DS18B20* section.)
5	4	1	GND	Ground

Overview

Figure 3 shows a block diagram of the DS18B20, and pin descriptions are given in the *Pin Description* table. The 64-bit ROM stores the device's unique serial code. The scratchpad memory contains the 2-byte temperature register that stores the digital output from the temperature sensor. In addition, the scratchpad provides access to the 1-byte upper and lower alarm trigger registers (T_H and T_L) and the 1-byte configuration register. The configuration register allows the user to set the resolution of the temperature-to-digital conversion to 9, 10, 11, or 12 bits. The T_H, T_L, and configuration registers are nonvolatile (EEPROM), so they will retain data when the device is powered down.

The DS18B20 uses Maxim's exclusive 1-Wire bus protocol that implements bus communication using one control signal. The control line requires a weak pullup resistor since all devices are linked to the bus via a 3-state or open-drain port (the DQ pin in the case of the DS18B20). In this bus system, the microprocessor (the master device) identifies and addresses devices on the bus using each device's unique 64-bit code. Because each device has a unique code, the number of devices that can be addressed on one bus is virtually unlimited. The 1-Wire bus protocol, including detailed explanations of the commands and "time slots," is covered in the *1-Wire Bus System* section.

Another feature of the DS18B20 is the ability to operate without an external power supply. Power is instead supplied through the 1-Wire pullup resistor through the DQ pin when the bus is high. The high bus signal also charges an internal capacitor (C_{PP}), which then supplies power to the device when the bus is low. This method of deriving power from the 1-Wire bus is referred to as "parasite power." As an alternative, the DS18B20 may also be powered by an external supply on V_{DD}.

Operation—Measuring Temperature

The core functionality of the DS18B20 is its direct-to-digital temperature sensor. The resolution of the temperature sensor is user-configurable to 9, 10, 11, or 12 bits, corresponding to increments of 0.5°C, 0.25°C, 0.125°C, and 0.0625°C, respectively. The default resolution at power-up is 12-bit. The DS18B20 powers up in a low-power idle state. To initiate a temperature measurement and A-to-D conversion, the master must issue a Convert T [44h] command. Following the conversion, the resulting thermal data is stored in the 2-byte temperature register in the scratchpad memory and the DS18B20 returns to its idle state. If the DS18B20 is powered by an external supply, the master can issue "read time slots" (see the *1-Wire Bus System* section) after the Convert T command and the DS18B20 will respond by transmitting 0 while the temperature conversion is in progress and 1 when the conversion is done. If the DS18B20 is powered with parasite power, this notification technique cannot be used since the bus must be pulled high by a strong pullup during the entire temperature conversion. The bus requirements for parasite power are explained in detail in the *Powering the DS18B20* section.

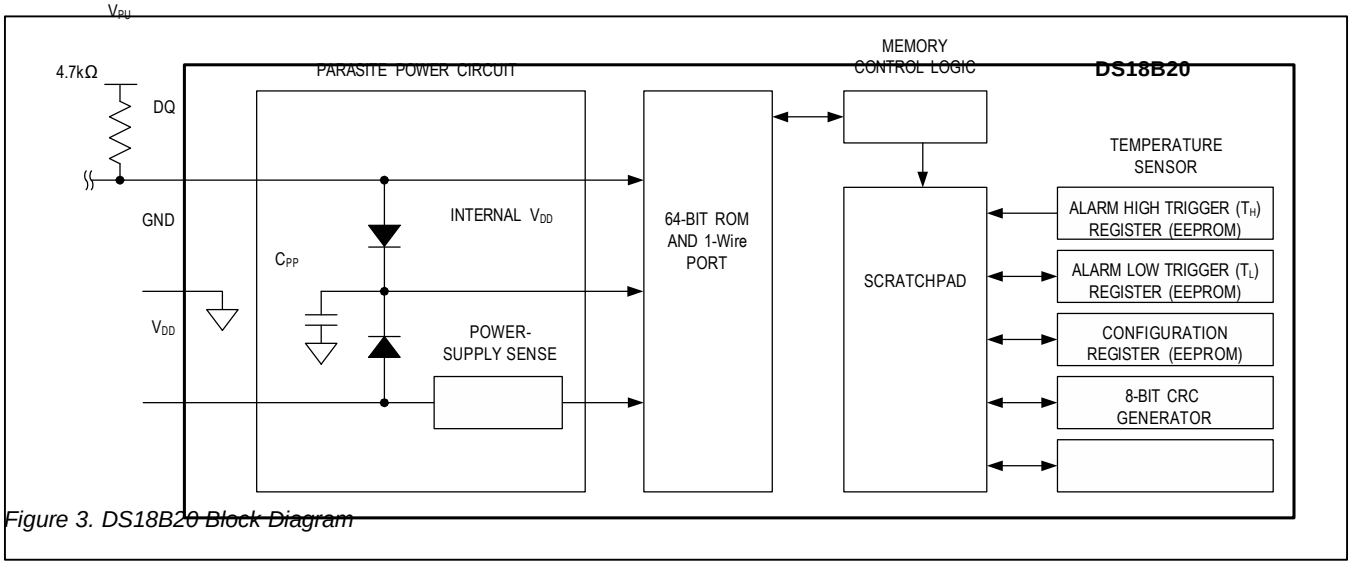


Figure 3. DS18B20 Block Diagram

The DS18B20 output temperature data is calibrated in degrees Celsius; for Fahrenheit applications, a lookup table or conversion routine must be used. The temperature data is stored as a 16-bit sign-extended two's complement number in the temperature register (see Figure 4). The sign bits (S) indicate if the temperature is positive or negative: for positive numbers S = 0 and for negative numbers S = 1. If the DS18B20 is configured for 12-bit resolution, all bits in the temperature register will contain valid data. For 11-bit resolution, bit 0 is undefined. For 10-bit resolution, bits 1 and 0 are undefined, and for 9-bit resolution bits 2, 1, and 0 are undefined. Table 1 gives examples of digital output data and the corresponding temperature reading for 12-bit resolution conversions.

Operation—Alarm Signaling

After the DS18B20 performs a temperature conversion, the temperature value is compared to the user-defined two's complement alarm trigger values stored in the 1-byte T_H and T_L registers (see Figure 5). The sign bit (S) indicates if the value is positive or negative: for positive numbers S = 0 and for negative numbers S = 1. The T_H and T_L registers are nonvolatile (EEPROM) so they will retain data when the device is powered down. T_H and T_L can be accessed through bytes 2 and 3 of the scratchpad as explained in the *Memory* section.

Only bits 11 through 4 of the temperature register are used in the T_H and T_L comparison since T_H and T_L are 8-bit registers. If the measured temperature is lower than

	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
LS BYTE	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}
	BIT 15	**BIT 14**	**BIT 13**	**BIT 12**	**BIT 11**	**BIT 10**	**BIT 9**	**BIT 8**
MS BYTE	S	S	S	S	S	2^6	2^5	2^4

S = SIGN

Figure 4. Temperature Register Format

Table 1. Temperature/Data Relationship

TEMPERATURE (°C)	DIGITAL OUTPUT (BINARY)	DIGITAL OUTPUT (HEX)
+125	0000 0111 1101 0000	07D0h
+85*	0000 0101 0101 0000	0550h
+25.0625	0000 0001 1001 0001	0191h
+10.125	0000 0000 1010 0010	00A2h
+0.5	0000 0000 0000 1000	0008h
0	0000 0000 0000 0000	0000h
-0.5	1111 1111 1111 1000	FFF8h
-10.125	1111 1111 0101 1110	FF5Eh
-25.0625	1111 1110 0110 1111	FE6Fh
-55	1111 1100 1001 0000	FC90h

**The power-on reset value of the temperature register is +85°C.*

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
S	2^6	2^5	2^4	2^3	2^2	2^1	2^0

Figure 5. T_H and T_L Register Format

or equal to T_L or higher than or equal to T_H, an alarm condition exists and an alarm flag is set inside the DS18B20. This flag is updated after every temperature measurement; therefore, if the alarm condition goes away, the flag will be turned off after the next temperature conversion.

The master device can check the alarm flag status of all DS18B20s on the bus by issuing an Alarm Search [ECh] command. Any DS18B20s with a set alarm flag will respond to the command, so the master can determine exactly which DS18B20s have experienced an alarm condition. If an alarm condition exists and the T_H or T_L settings have changed, another temperature conversion should be done to validate the alarm condition.

Powering the DS18B20

The DS18B20 can be powered by an external supply on the V_{DD} pin, or it can operate in "parasite power" mode, which allows the DS18B20 to function without a local external supply. Parasite power is very useful for applications that require remote temperature sensing or that are very space constrained. Figure 3 shows the DS18B20's parasite-power control circuitry, which "steals" power from the 1-Wire bus via the DQ pin when the bus is high. The stolen charge powers the DS18B20 while the bus is high, and some of the charge is stored on the parasite power capacitor (C_{PP}) to provide power when the bus is low. When the DS18B20 is used in parasite power mode, the V_{DD} pin must be connected to ground.

In parasite power mode, the 1-Wire bus and CPP can provide sufficient current to the DS18B20 for most operations as long as the specified timing and voltage requirements are met (see the *DC Electrical Characteristics* and *AC Electrical Characteristics*). However, when the DS18B20 is performing temperature conversions or copying data from the scratchpad memory to EEPROM, the operating current can be as high as 1.5mA. This current can cause an unacceptable voltage drop across the weak 1-Wire pullup resistor and is more current than can be supplied by C_{PP}. To assure that the DS18B20 has sufficient supply current, it is necessary to provide a strong pullup on the 1-Wire bus whenever temperature conversions are taking place or data is being copied from the scratchpad to EEPROM. This can be accomplished by using a MOSFET to pull the bus directly to the rail as shown in Figure 6. The 1-Wire bus must be switched to the strong pullup within 10µs (max) after a Convert T [44h] or Copy Scratchpad [48h] command is issued, and the bus must be held high by the pullup for the duration of the conversion (t_{CONV}) or data transfer (t_{WR} = 10ms). No other activity can take place on the 1-Wire bus while the pullup is enabled.

The DS18B20 can also be powered by the conventional method of connecting an external power supply to the V_{DD} pin, as shown in Figure 7. The advantage of this method is that the MOSFET pullup is not required, and the 1-Wire bus is free to carry other traffic during the temperature conversion time.

The use of parasite power is not recommended for temperatures above +100°C since the DS18B20 may not be able to sustain communications due to the higher leakage currents that can exist at these temperatures. For applications in which such temperatures are likely, it is strongly recommended that the DS18B20 be powered by an external power supply.

In some situations the bus master may not know whether the DS18B20s on the bus are parasite powered or powered by external supplies. The master needs this information to determine if the strong bus pullup should be used during temperature conversions. To get this information, the master can issue a Skip ROM [CCh] command followed by a Read Power Supply [B4h] command followed by a "read time slot". During the read time slot, parasite powered DS18B20s will pull the bus low, and externally powered DS18B20s will let the bus remain high. If the bus is pulled low, the master knows that it must supply the strong pullup on the 1-Wire bus during temperature conversions.

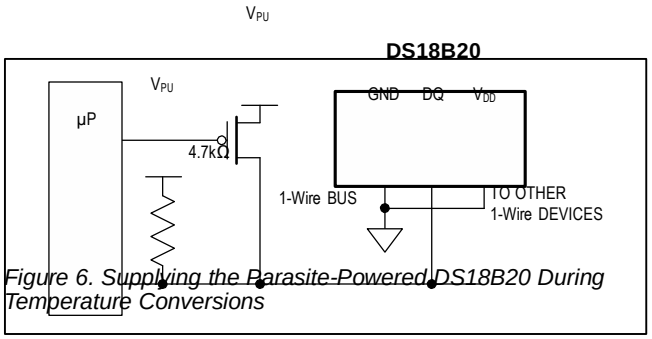


Figure 6. Supplying the Parasite-Powered DS18B20 During Temperature Conversions

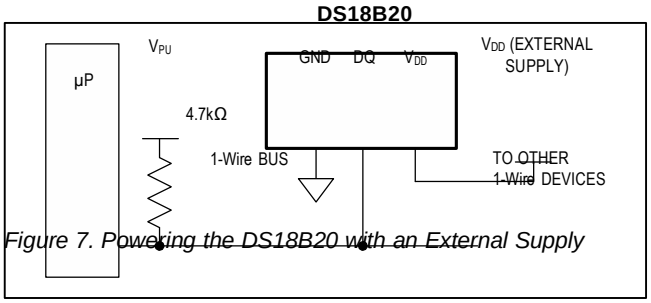


Figure 7. Powering the DS18B20 with an External Supply

64-BIT Lasered ROM code

Each DS18B20 contains a unique 64–bit code (see Figure 8) stored in ROM. The least significant 8 bits of the ROM code contain the DS18B20's 1-Wire family code: 28h. The next 48 bits contain a unique serial number. The most significant 8 bits contain a cyclic redundancy check (CRC) byte that is calculated from the first 56 bits of the ROM code. A detailed explanation of the CRC bits is provided in the *CRC Generation* section. The 64-bit ROM code and associated ROM function control logic allow the DS18B20 to operate as a 1-Wire device using the protocol detailed in the *1-Wire Bus System* section.

Memory

The DS18B20's memory is organized as shown in Figure 9. The memory consists of an SRAM scratchpad with nonvolatile EEPROM storage for the high and low alarm trigger registers (T_H and T_L) and configuration register. Note that if the DS18B20 alarm function is not used, the TH and TL registers can serve as general-purpose memory. All memory commands are described in detail in the *DS18B20 Function Commands* section.

Byte 0 and byte 1 of the scratchpad contain the LSB and the MSB of the temperature register, respectively. These bytes are read-only. Bytes 2 and 3 provide access to TH and TL registers. Byte 4 contains the configuration register data, which is explained in detail in the *Configuration Register* section. Bytes 5, 6, and 7 are reserved for internal use by the device and cannot be overwritten.

Byte 8 of the scratchpad is read-only and contains the CRC code for bytes 0 through 7 of the scratchpad. The DS18B20 generates this CRC using the method described in the *CRC Generation* section.

Data is written to bytes 2, 3, and 4 of the scratchpad using the Write Scratchpad [4Eh] command; the data must be transmitted to the DS18B20 starting with the least significant bit of byte 2. To verify data integrity, the scratchpad can be read (using the Read Scratchpad [BEh] command) after the data is written. When reading the scratchpad, data is transferred over the 1-Wire bus starting with the least significant bit of byte 0. To transfer the T_H, T_L and configuration data from the scratchpad to EEPROM, the master must issue the Copy Scratchpad [48h] command.

Data in the EEPROM registers is retained when the device is powered down; at power-up the EEPROM data is reloaded into the corresponding scratchpad locations. Data can also be reloaded from EEPROM to the scratchpad at any time using the Recall E^2 [B8h] command. The master can issue read time slots following the Recall E^2 command and the DS18B20 will indicate the status of the recall by transmitting 0 while the recall is in progress and 1 when the recall is done.

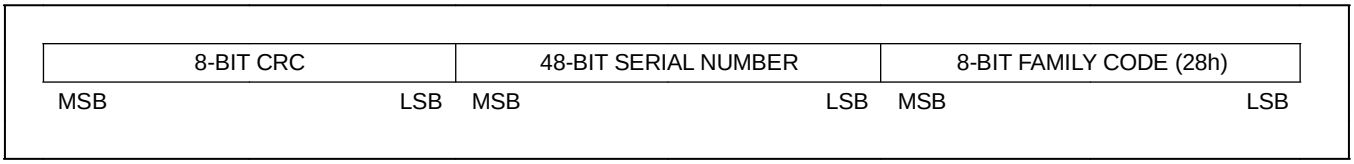


Figure 8. 64-Bit Lasered ROM Code

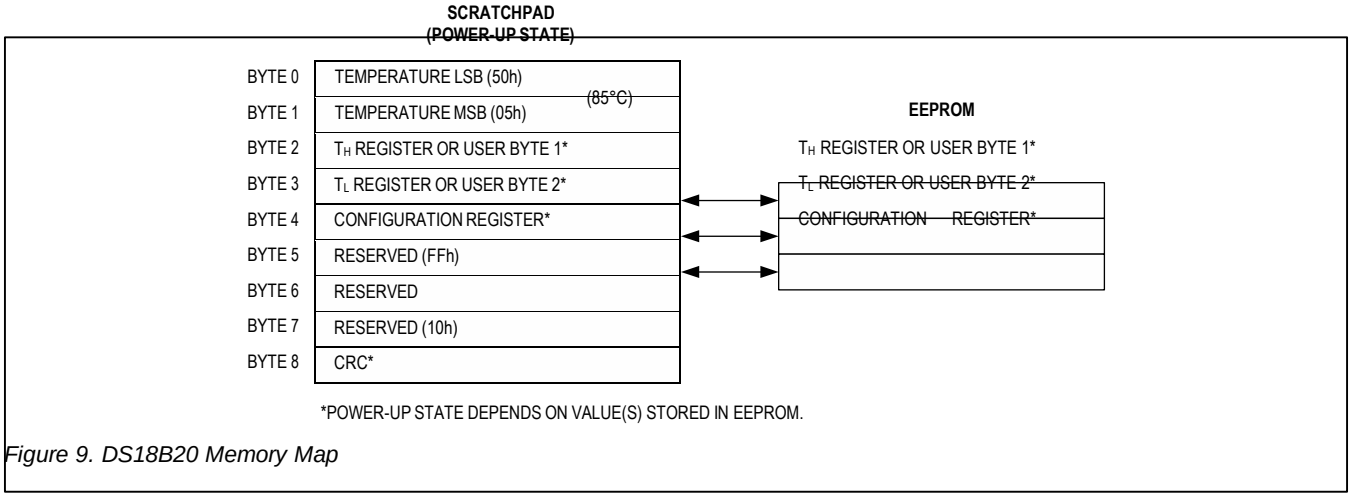


Figure 9. DS18B20 Memory Map

Configuration Register

Byte 4 of the scratchpad memory contains the configuration register, which is organized as illustrated in Figure 10. The user can set the conversion resolution of the DS18B20 using the R0 and R1 bits in this register as shown in Table 2. The power-up default of these bits is R0 = 1 and R1 = 1 (12-bit resolution). Note that there is a direct tradeoff between resolution and conversion time. Bit 7 and bits 0 to 4 in the configuration register are reserved for internal use by the device and cannot be overwritten.

CRC Generation

CRC bytes are provided as part of the DS18B20's 64-bit ROM code and in the 9th byte of the scratchpad memory. The ROM code CRC is calculated from the first 56 bits of the ROM code and is contained in the most significant byte of the ROM. The scratchpad CRC is calculated from the data stored in the scratchpad, and therefore it changes when the data in the scratchpad changes. The CRCs provide the bus master with a method of data validation when data is read from the DS18B20. To verify that data has been read correctly, the bus master must re-calculate the CRC from the received data and then compare this value to either the ROM code CRC (for ROM reads) or to the scratchpad CRC (for scratchpad reads). If the calculated CRC matches the read CRC, the data has been received error free. The comparison of CRC values and the decision to continue with an operation are determined entirely by the bus master. There is no circuitry inside the DS18B20 that prevents a command sequence from proceeding if the DS18B20 CRC (ROM or scratchpad) does not match the value generated by the bus master.

The equivalent polynomial function of the CRC (ROM or scratchpad) is:

$$CRC = X^8 + X^5 + X^4 + 1$$

The bus master can re-calculate the CRC and compare it to the CRC values from the DS18B20 using the polynomial generator shown in Figure 11. This circuit consists of a shift register and XOR gates, and the shift register bits are initialized to 0. Starting with the least significant bit of the ROM code or the least significant bit of byte 0 in the scratchpad, one bit at a time should shifted into the shift register. After shifting in the 56th bit from the ROM or the most significant bit of byte 7 from the scratchpad, the polynomial generator will contain the recalculated CRC. Next, the 8-bit ROM code or scratchpad CRC from the DS18B20 must be shifted into the circuit. At this point, if the re-calculated CRC was correct, the shift register will contain all 0s. Additional information about the Maxim 1-Wire cyclic redundancy check is available in *Application Note 27: Understanding and Using Cyclic Redundancy Checks with Maxim iButton Products*.

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
0	R1	R0	1	1	1	1	1

Figure 10. Configuration Register

Table 2. Thermometer Resolution Configuration

R1	R0	RESOLUTION (BITS)	MAX CONVERSION TIME	
0	0	9	93.75ms	(t_{CONV}/8)
0	1	10	187.5ms	(t_{CONV}/4)
1	0	11	375ms	(t_{CONV}/2)
1	1	12	750ms	(t_{CONV})

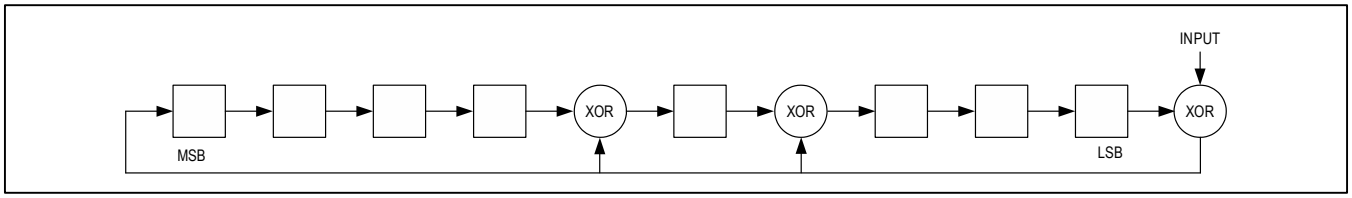


Figure 11. CRC Generator

1-Wire Bus System

The 1-Wire bus system uses a single bus master to control one or more slave devices. The DS18B20 is always a slave. When there is only one slave on the bus, the system is referred to as a “single-drop” system; the system is “multidrop” if there are multiple slaves on the bus.

All data and commands are transmitted least significant bit first over the 1-Wire bus.

The following discussion of the 1-Wire bus system is broken down into three topics: hardware configuration, transaction sequence, and 1-Wire signaling (signal types and timing).

Hardware Configuration

The 1-Wire bus has by definition only a single data line. Each device (master or slave) interfaces to the data line via an open-drain or 3-state port. This allows each device to “release” the data line when the device is not transmitting data so the bus is available for use by another device. The 1-Wire port of the DS18B20 (the DQ pin) is open drain with an internal circuit equivalent to that shown in Figure 12.

The 1-Wire bus requires an external pullup resistor of approximately 5kΩ; thus, the idle state for the 1-Wire bus is high. If for any reason a transaction needs to be suspended, the bus MUST be left in the idle state if the transaction is to resume. Infinite recovery time can occur between bits so long as the 1-Wire bus is in the inactive (high) state during the recovery period. If the bus is held low for more than 480µs, all components on the bus will be reset.

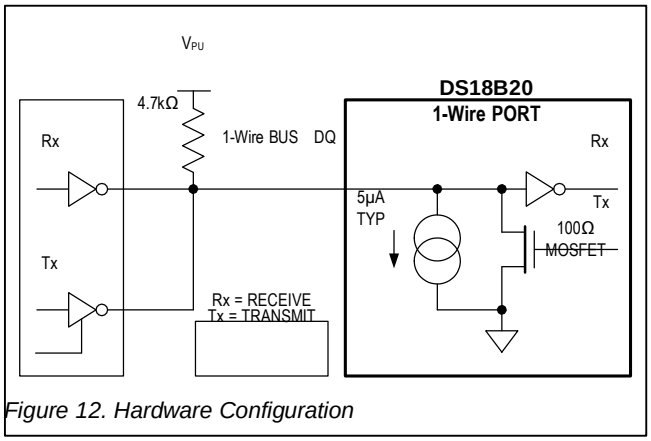


Figure 12. Hardware Configuration

Transaction Sequence

The transaction sequence for accessing the DS18B20 is as follows:

Step 1. Initialization

Step 2. ROM Command (followed by any required data exchange)

Step 3. DS18B20 Function Command (followed by any required data exchange)

It is very important to follow this sequence every time the DS18B20 is accessed, as the DS18B20 will not respond if any steps in the sequence are missing or out of order. Exceptions to this rule are the Search ROM [F0h] and Alarm Search [ECh] commands. After issuing either of these ROM commands, the master must return to Step 1 in the sequence.

Initialization

All transactions on the 1-Wire bus begin with an initialization sequence. The initialization sequence consists of a reset pulse transmitted by the bus master followed by presence pulse(s) transmitted by the slave(s). The presence pulse lets the bus master know that slave devices (such as the DS18B20) are on the bus and are ready to operate. Timing for the reset and presence pulses is detailed in the *1-Wire Signaling* section.

ROM Commands

After the bus master has detected a presence pulse, it can issue a ROM command. These commands operate on the unique 64-bit ROM codes of each slave device and allow the master to single out a specific device if many are present on the 1-Wire bus. These commands also allow the master to determine how many and what types of devices are present on the bus or if any device has experienced an alarm condition. There are five ROM commands, and each command is 8 bits long. The master device must issue an appropriate ROM command before issuing a DS18B20 function command. A flowchart for operation of the ROM commands is shown in Figure 13.

Search Rom [F0h]

When a system is initially powered up, the master must identify the ROM codes of all slave devices on the bus, which allows the master to determine the number of slaves and their device types. The master learns the ROM codes through a process of elimination that requires the master to perform a Search ROM cycle (i.e., Search ROM command followed by data exchange) as many times as necessary to identify all of the slave devices.

If there is only one slave on the bus, the simpler Read ROM [33h] command can be used in place of the Search ROM process. For a detailed explanation of the Search ROM procedure, refer to *Application Note 937: Book of iButton® Standards*. After every Search ROM cycle, the bus master must return to Step 1 (Initialization) in the transaction sequence.

Read Rom [33h]

This command can only be used when there is one slave on the bus. It allows the bus master to read the slave's 64-bit ROM code without using the Search ROM procedure. If this command is used when there is more than one slave present on the bus, a data collision will occur when all the slaves attempt to respond at the same time.

Match Rom [55H]

The match ROM command followed by a 64-bit ROM code sequence allows the bus master to address a specific slave device on a multidrop or single-drop bus. Only the slave that exactly matches the 64-bit ROM code sequence will respond to the function command issued by the master; all other slaves on the bus will wait for a reset pulse.

Skip Rom [CCh]

The master can use this command to address all devices on the bus simultaneously without sending out any ROM code information. For example, the master can make all DS18B20s on the bus perform simultaneous temperature conversions by issuing a Skip ROM command followed by a Convert T [44h] command.

Note that the Read Scratchpad [BEh] command can follow the Skip ROM command only if there is a single slave device on the bus. In this case, time is saved by allowing the master to read from the slave without sending the device's 64-bit ROM code. A Skip ROM command followed by a Read Scratchpad command will cause a data collision on the bus if there is more than one slave since multiple devices will attempt to transmit data simultaneously.

Alarm Search [ECh]

The operation of this command is identical to the operation of the Search ROM command except that only slaves with a set alarm flag will respond. This command allows the master device to determine if any DS18B20s experienced an alarm condition during the most recent temperature conversion. After every Alarm Search cycle (i.e., Alarm Search command followed by data exchange), the bus master must return to Step 1 (Initialization) in the transaction sequence. See the *Operation—Alarm Signaling* section for an explanation of alarm flag operation.

DS18B20 Function Commands

After the bus master has used a ROM command to address the DS18B20 with which it wishes to communicate, the master can issue one of the DS18B20 function commands. These commands allow the master to write to and read from the DS18B20's scratchpad memory, initiate temperature conversions and determine the power supply mode. The DS18B20 function commands, which are described below, are summarized in Table 3 and illustrated by the flowchart in Figure 14.

Convert T [44h]

This command initiates a single temperature conversion. Following the conversion, the resulting thermal data is stored in the 2-byte temperature register in the scratchpad memory and the DS18B20 returns to its low-power idle state. If the device is being used in parasite power mode, within 10µs (max) after this command is issued the master must enable a strong pullup on the 1-Wire bus for the duration of the conversion (t_{CONV}) as described in the *Powering the DS18B20* section. If the DS18B20 is powered by an external supply, the master can issue read time slots after the Convert T command and the DS18B20 will respond by transmitting a 0 while the temperature conversion is in progress and a 1 when the conversion is done. In parasite power mode this notification technique cannot be used since the bus is pulled high by the strong pullup during the conversion.

Write Scratchpad [4Eh]

This command allows the master to write 3 bytes of data to the DS18B20's scratchpad. The first data byte is written into the T_H register (byte 2 of the scratchpad), the second byte is written into the T_L register (byte 3), and the third byte is written into the configuration register (byte 4). Data must be transmitted least significant bit first. All three bytes MUST be written before the master issues a reset, or the data may be corrupted.

Read Scratchpad [BEh]

This command allows the master to read the contents of the scratchpad. The data transfer starts with the least significant bit of byte 0 and continues through the scratchpad until the 9th byte (byte 8 – CRC) is read. The master may issue a reset to terminate reading at any time if only part of the scratchpad data is needed.

iButton is a registered trademark of Maxim Integrated Products, Inc.

Copy Scratchpad [48h]

This command copies the contents of the scratchpad T_H, T_L and configuration registers (bytes 2, 3 and 4) to EEPROM. If the device is being used in parasite power mode, within 10µs (max) after this command is issued the master must enable a strong pullup on the 1-Wire bus for at least 10ms as described in the *Powering the DS18B20* section.

Recall E^2 [B8h]

This command recalls the alarm trigger values (T_H and T_L) and configuration data from EEPROM and places the data in bytes 2, 3, and 4, respectively, in the scratchpad memory. The master device can issue read time slots following the Recall E^2 command and the DS18B20 will indicate the status of the recall by transmitting 0 while the recall is in progress and 1 when the recall is done. The recall operation happens automatically at power-up, so valid data is available in the scratchpad as soon as power is applied to the device.

Read Power Supply [B4h]

The master device issues this command followed by a read time slot to determine if any DS18B20s on the bus are using parasite power. During the read time slot, parasite powered DS18B20s will pull the bus low, and externally powered DS18B20s will let the bus remain high. See the *Powering the DS18B20* section for usage information for this command.

Table 3. DS18B20 Function Command Set

COMMAND	DESCRIPTION	PROTOCOL	1-Wire BUS ACTIVITY AFTER COMMAND IS ISSUED	NOTES
TEMPERATURE CONVERSION COMMANDS				
Convert T	Initiates temperature conversion.	44h	DS18B20 transmits conversion status to master (not applicable for parasite-powered DS18B20s).	1
MEMORY COMMANDS				
Read Scratchpad	Reads the entire scratchpad including the CRC byte.	BEh	DS18B20 transmits up to 9 data bytes to master.	2
Write Scratchpad	Writes data into scratchpad bytes 2, 3, and 4 (T_H, T_L, and configuration registers).	4Eh	Master transmits 3 data bytes to DS18B20.	3
Copy Scratchpad	Copies T_H, T_L, and configuration register data from the scratchpad to EEPROM.	48h	None	1
Recall E^2	Recalls T_H, T_L, and configuration register data from EEPROM to the scratchpad.	B8h	DS18B20 transmits recall status to master.	
Read Power Supply	Signals DS18B20 power supply mode to the master.	B4h	DS18B20 transmits supply status to master.	

Note 1: For parasite-powered DS18B20s, the master must enable a strong pullup on the 1-Wire bus during temperature conversions and copies from the scratchpad to EEPROM. No other bus activity may take place during this time.
Note 2: The master can interrupt the transmission of data at any time by issuing a reset.
Note 3: All three bytes must be written before a reset is issued.

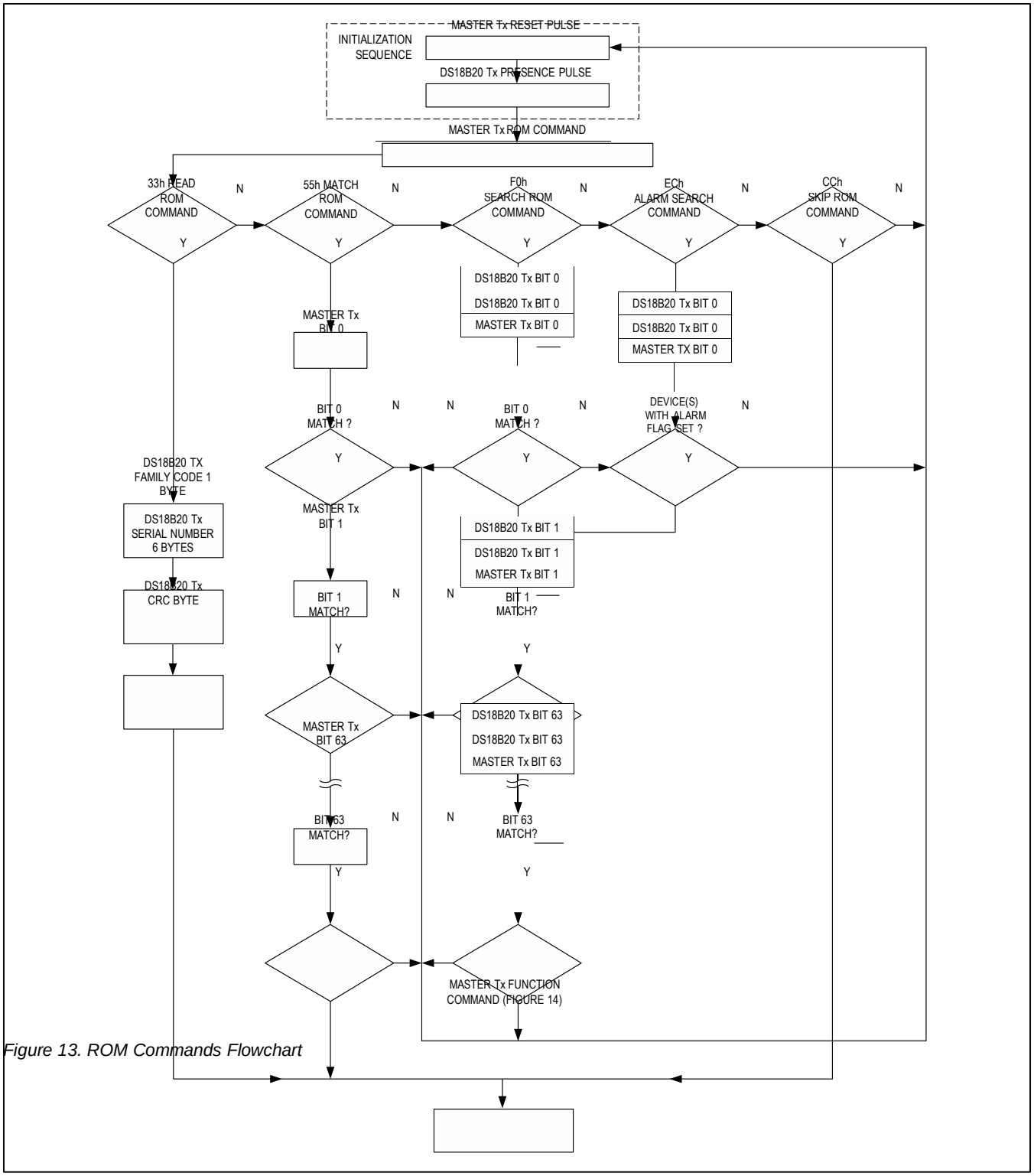


Figure 13. ROM Commands Flowchart

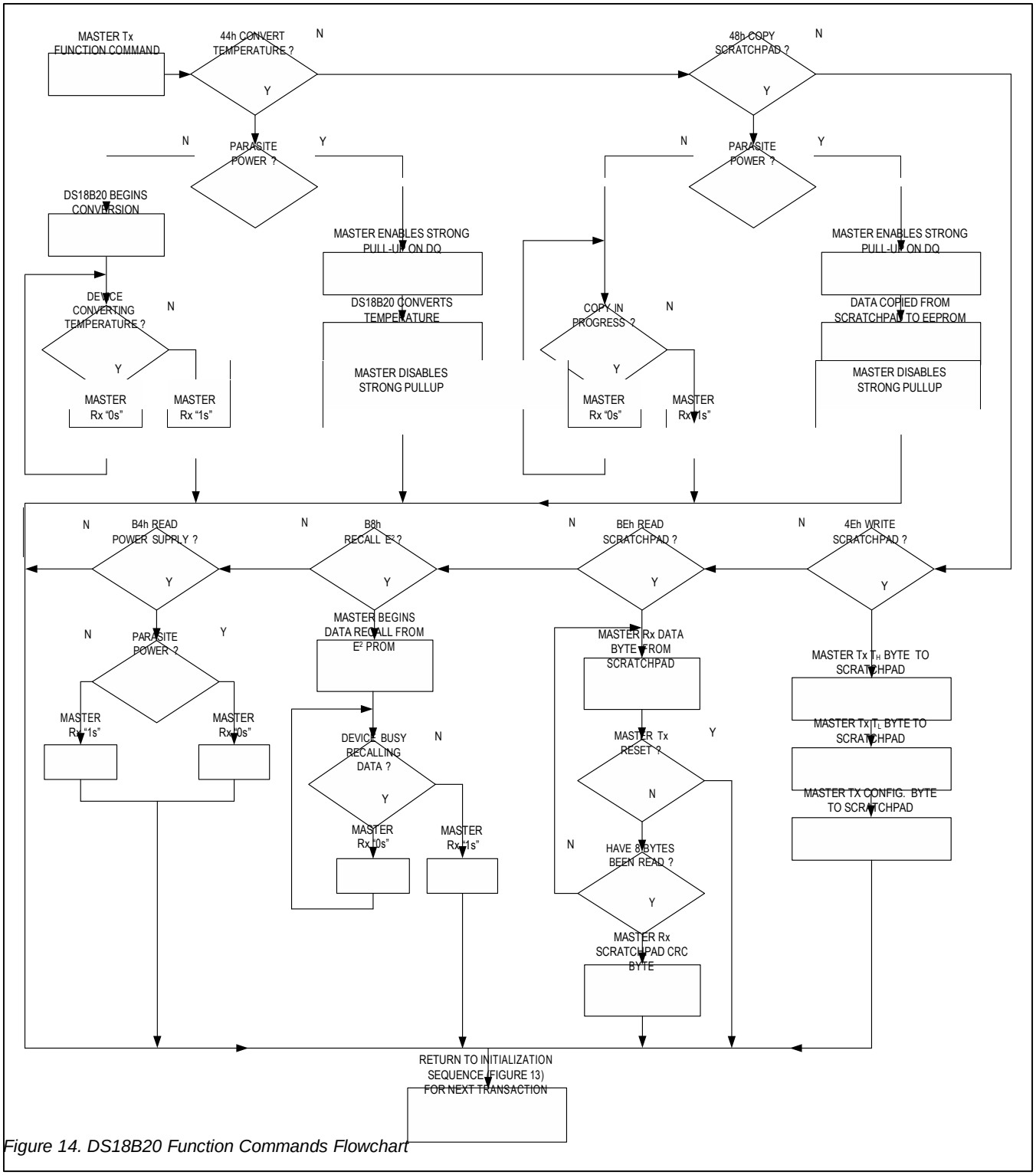


Figure 14. DS18B20 Function Commands Flowchart

1-Wire Signaling

The DS18B20 uses a strict 1-Wire communication protocol to ensure data integrity. Several signal types are defined by this protocol: reset pulse, presence pulse, write 0, write 1, read 0, and read 1. The bus master initiates all these signals, with the exception of the presence pulse.

Initialization Procedure—Reset And Presence Pulses

All communication with the DS18B20 begins with an initialization sequence that consists of a reset pulse from the master followed by a presence pulse from the DS18B20. This is illustrated in Figure 15. When the DS18B20 sends the presence pulse in response to the reset, it is indicating to the master that it is on the bus and ready to operate.

During the initialization sequence the bus master transmits (T_X) the reset pulse by pulling the 1-Wire bus low for a minimum of 480µs. The bus master then releases the bus and goes into receive mode (R_X). When the bus is released, the 5kΩ pullup resistor pulls the 1-Wire bus high. When the DS18B20 detects this rising edge, it waits 15µs to 60µs and then transmits a presence pulse by pulling the 1-Wire bus low for 60µs to 240µs.

Read/Write Time Slots

The bus master writes data to the DS18B20 during write time slots and reads data from the DS18B20 during read time slots. One bit of data is transmitted over the 1-Wire bus per time slot.

Write Time Slots

There are two types of write time slots: “Write 1” time slots and “Write 0” time slots. The bus master uses a Write 1 time slot to write a logic 1 to the DS18B20 and a Write 0 time slot to write a logic 0 to the DS18B20. All write time slots must be a minimum of 60µs in duration with a minimum of a 1µs recovery time between individual write slots. Both types of write time slots are initiated by the master pulling the 1-Wire bus low (see Figure 14).

To generate a Write 1 time slot, after pulling the 1-Wire bus low, the bus master must release the 1-Wire bus within 15µs. When the bus is released, the 5kΩ pullup resistor will pull the bus high. To generate a Write 0 time slot, after pulling the 1-Wire bus low, the bus master must continue to hold the bus low for the duration of the time slot (at least 60µs).

The DS18B20 samples the 1-Wire bus during a window that lasts from 15µs to 60µs after the master initiates the write time slot. If the bus is high during the sampling window, a 1 is written to the DS18B20. If the line is low, a 0 is written to the DS18B20.

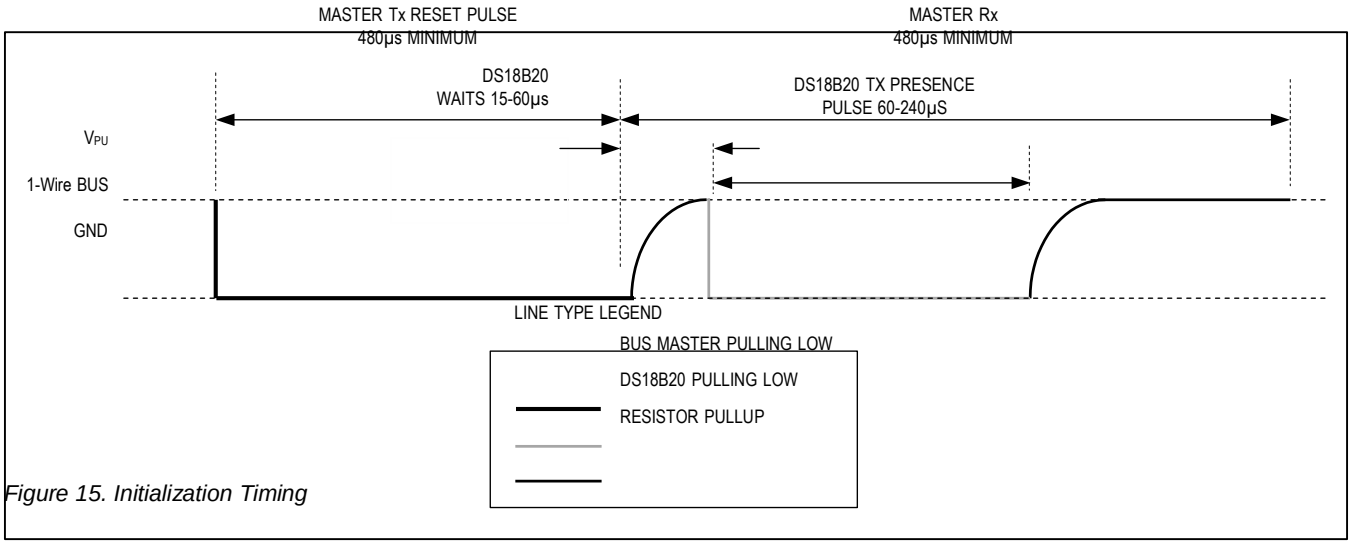


Figure 15. Initialization Timing

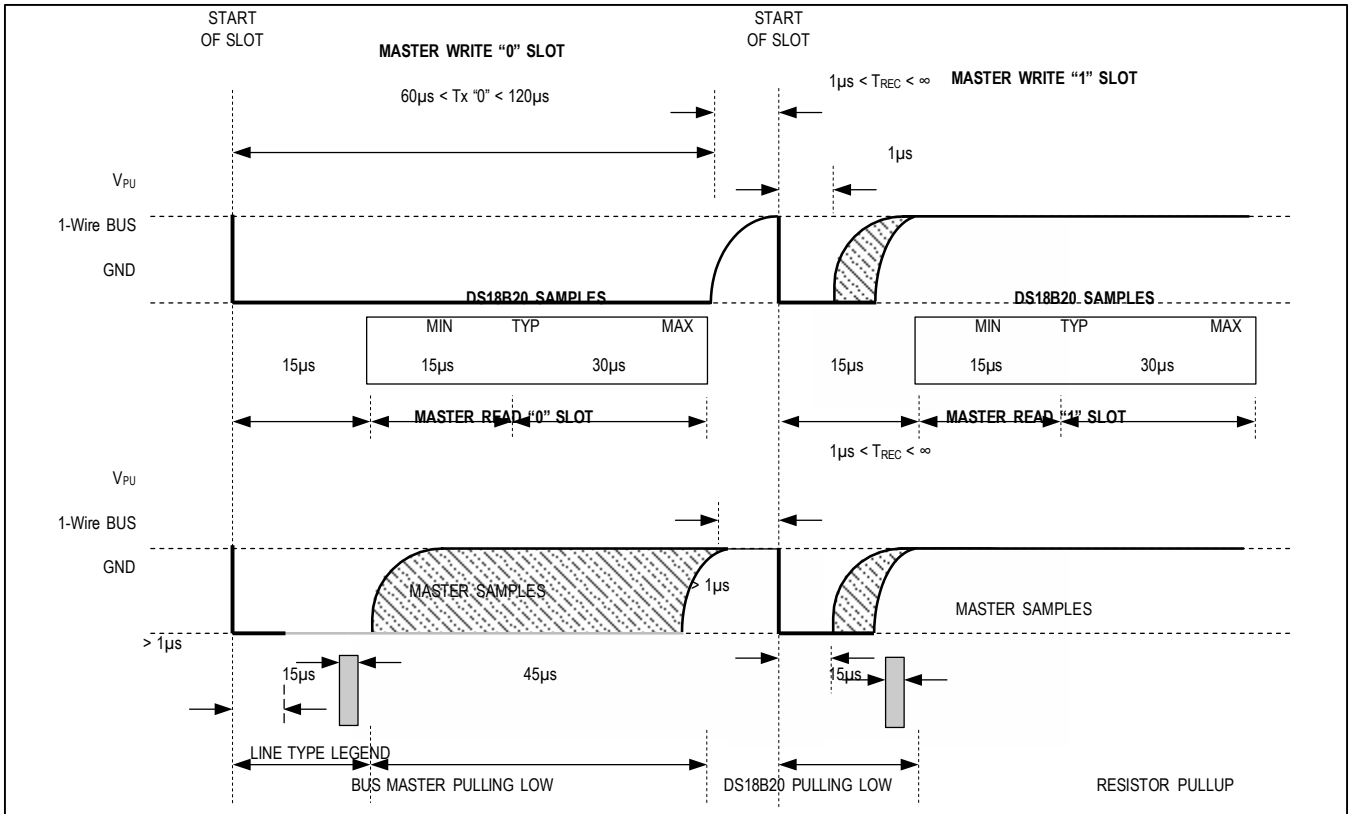


Figure 16. Read/Write Time Slot Timing Diagram

Read Time Slots

The DS18B20 can only transmit data to the master when the master issues read time slots. Therefore, the master must generate read time slots immediately after issuing a Read Scratchpad [BEh] or Read Power Supply [B4h] command, so that the DS18B20 can provide the requested data. In addition, the master can generate read time slots after issuing Convert T [44h] or Recall E^2 [B8h] commands to find out the status of the operation as explained in the *DS18B20 Function Commands* section.

All read time slots must be a minimum of 60µs in duration with a minimum of a 1µs recovery time between slots. A read time slot is initiated by the master device pulling the 1-Wire bus low for a minimum of 1µs and then releasing the bus (see Figure 16). After the master initiates the read time slot, the DS18B20 will begin transmitting a 1 or 0 on bus. The DS18B20 transmits a 1 by leaving the bus high and transmits a 0 by pulling the bus low. When transmitting a 0, the DS18B20 will release the bus by the end of the time slot, and the bus will be pulled back to its high idle state by the pullup resister. Output data from the DS18B20 is valid for 15µs after the falling edge that initiated the read time slot. Therefore, the master must release the bus and then sample the bus state within 15µs from the start of the slot.

Figure 17 illustrates that the sum of T_{INIT}, T_{RC}, and T_{SAMPLE} must be less than 15µs for a read time slot. Figure 18 shows that system timing margin is maximized by keeping T_{INIT} and T_{RC} as short as possible and by locating the master sample time during read time slots towards the end of the 15µs period.

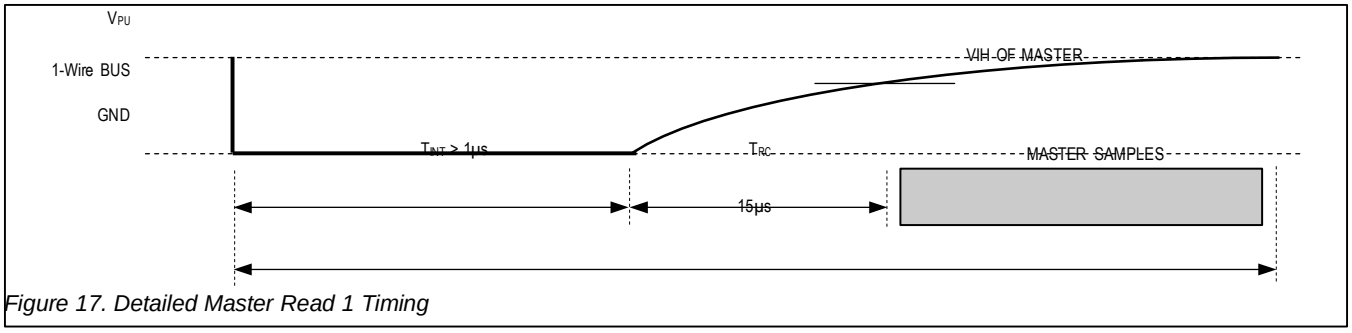


Figure 17. Detailed Master Read 1 Timing

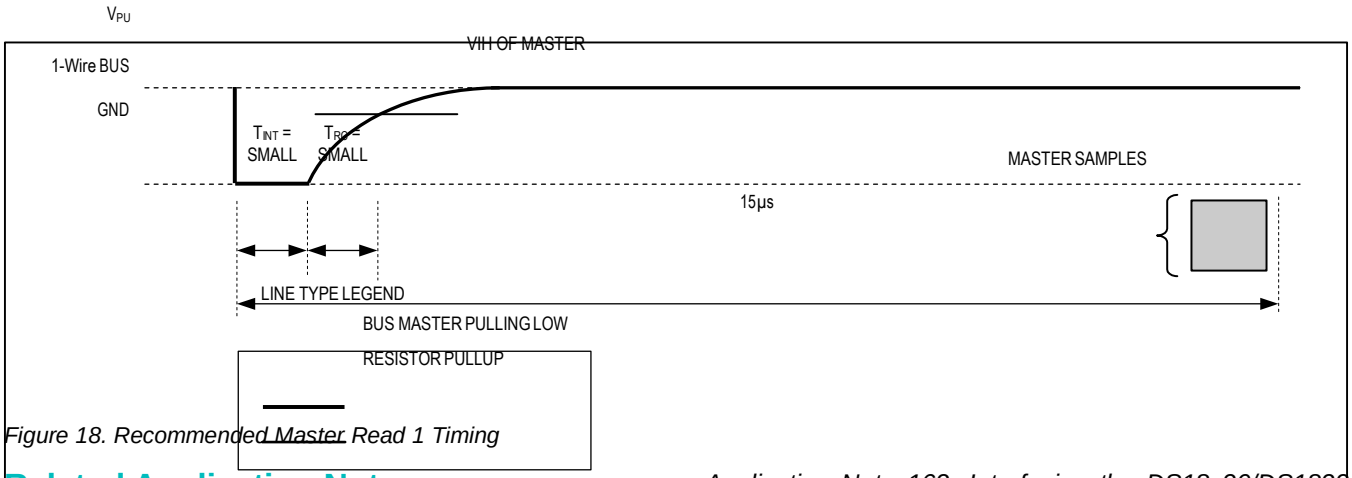


Figure 18. Recommended Master Read 1 Timing

Related Application Notes

The following application notes can be applied to the DS18B20 and are available at www.maximintegrated.com.

Application Note 27: Understanding and Using Cyclic Redundancy Checks with Maxim iButton Products

Application Note 122: Using Dallas' 1-Wire ICs in 1-Cell Li-Ion Battery Packs with Low-Side N-Channel Safety FETs Master

Application Note 126: 1-Wire Communication Through Software

Application Note 162: Interfacing the DS18x20/DS1822 1-Wire Temperature Sensor in a Microcontroller Environment

Application Note 208: Curve Fitting the Error of a Bandgap-Based Digital Temperature Sensor

Application Note 2420: 1-Wire Communication with a Microchip PICmicro Microcontroller

Application Note 3754: Single-Wire Serial Bus Carries Isolated Power and Data

Sample 1-Wire subroutines that can be used in conjunction with *Application Note 74: Reading and Writing iButtons via Serial Interfaces* can be downloaded from the Maxim website.

DS18B20 Operation Example 1

In this example there are multiple DS18B20s on the bus and they are using parasite power. The bus master initiates a temperature conversion in a specific DS18B20 and then reads its scratchpad and recalculates the CRC to verify the data.

MASTER MODE	DATA (LSB FIRST)	COMMENTS
Tx	Reset	Master issues reset pulse.
Rx	Presence	DS18B20s respond with presence pulse.
Tx	55h	Master issues Match ROM command.
Tx	64-bit ROM code	Master sends DS18B20 ROM code.
Tx	44h	Master issues Convert T command.
Tx	DQ line held high by strong pullup	Master applies strong pullup to DQ for the duration of the conversion (t_{CONV}).
Tx	Reset	Master issues reset pulse.
Rx	Presence	DS18B20s respond with presence pulse.
Tx	55h	Master issues Match ROM command.
Tx	64-bit ROM code	Master sends DS18B20 ROM code.
Tx	BEh	Master issues Read Scratchpad command.
Rx	9 data bytes	Master reads entire scratchpad including CRC. The master then recalculates the CRC of the first eight data bytes from the scratchpad and compares the calculated CRC with the read CRC (byte 9). If they match, the master continues; if not, the read operation is repeated.

DS18B20 Operation Example 2

In this example there is only one DS18B20 on the bus and it is using parasite power. The master writes to the TH, TL, and configuration registers in the DS18B20 scratchpad and then reads the scratchpad and recalculates the CRC to verify the data. The master then copies the scratchpad contents to EEPROM.

MASTER MODE	DATA (LSB FIRST)	COMMENTS
Tx	Reset	Master issues reset pulse.
Rx	Presence	DS18B20 responds with presence pulse.
Tx	CCh	Master issues Skip ROM command.
Tx	4Eh	Master issues Write Scratchpad command.
Tx	3 data bytes	Master sends three data bytes to scratchpad (T_H, T_L, and config).
Tx	Reset	Master issues reset pulse.
Rx	Presence	DS18B20 responds with presence pulse.
Tx	CCh	Master issues Skip ROM command.
Tx	BEh	Master issues Read Scratchpad command.
Rx	9 data bytes	Master reads entire scratchpad including CRC. The master then recalculates the CRC of the first eight data bytes from the scratchpad and compares the calculated CRC with the read CRC (byte 9). If they match, the master continues; if not, the read operation is repeated.
Tx	Reset	Master issues reset pulse.
Rx	Presence	DS18B20 responds with presence pulse.
Tx	CCh	Master issues Skip ROM command.
Tx	48h	Master issues Copy Scratchpad command.
Tx	DQ line held high by strong pullup	Master applies strong pullup to DQ for at least 10ms while copy operation is in progress.

Ordering Information

PART	TEMP RANGE	PIN-PACKAGE	TOP MARK
DS18B20	-55°C to +125°C	3 TO-92	18B20
DS18B20+	-55°C to +125°C	3 TO-92	18B20
DS18B20/T&R	-55°C to +125°C	3 TO-92 (2000 Piece)	18B20
DS18B20+T&R	-55°C to +125°C	3 TO-92 (2000 Piece)	18B20
DS18B20-SL/T&R	-55°C to +125°C	3 TO-92 (2000 Piece)*	18B20
DS18B20-SL+T&R	-55°C to +125°C	3 TO-92 (2000 Piece)*	18B20
DS18B20U	-55°C to +125°C	8 FSOP	18B20
DS18B20U+	-55°C to +125°C	8 FSOP	18B20
DS18B20U/T&R	-55°C to +125°C	8 FSOP (3000 Piece)	18B20
DS18B20U+T&R	-55°C to +125°C	8 FSOP (3000 Piece)	18B20
DS18B20Z	-55°C to +125°C	8 SO	DS18B20
DS18B20Z+	-55°C to +125°C	8 SO	DS18B20
DS18B20Z/T&R	-55°C to +125°C	8 SO (2500 Piece)	DS18B20
DS18B20Z+T&R	-55°C to +125°C	8 SO (2500 Piece)	DS18B20

+Denotes a lead-free package. A "+" will appear on the top mark of lead-free packages.

T&R = Tape and reel.

**TO-92 packages in tape and reel can be ordered with straight or formed leads. Choose "SL" for straight leads. Bulk TO-92 orders are straight leads only.*

Revision History

REVISION DATE	DESCRIPTION	PAGES CHANGED
030107	In the Absolute Maximum Ratings section, removed the reflow oven temperature value of +220°C. Reference to JEDEC specification for reflow remains.	19
101207	In the *Operation—Alarm Signaling* section, added “or equal to” in the description for a TH alarm condition	5
	In the *Memory* section, removed incorrect text describing memory.	7
	In the *Configuration Register* section, removed incorrect text describing configuration register.	8
042208	In the *Ordering Information* table, added TO-92 straight-lead packages and included a note that the TO-92 package in tape and reel can be ordered with either formed or straight leads.	2
1/15	Updated *Benefits and Features* section	1

PH meter(SKU: SEN0161)

Contents [hide]

Analog pH Meter Kit

Introduction

Need to measure water quality and other parameters but haven't got any low cost pH meter? Find it difficult to use with Arduino? Here comes an analog pH meter, specially designed for **Arduino controllers** and has built-in simple, convenient and practical connection and features. It has an LED which works as the Power Indicator, a BNC connector and PH2.0 sensor interface. To use it, just connect the pH sensor with BNC connector, and plug the PH2.0 interface into the analog input port of any Arduino controller. If pre-programmed, you will get the pH value easily. Comes in compact plastic box with foams for better mobile storage.

Attention:In order to ensure the accuracy of the pH probe, you need to use the standard solution to calibrate it regularly.Generally, the period is about half a year. If you meaure the dirty aqueous solution, you need to increase the frequency of calibration.

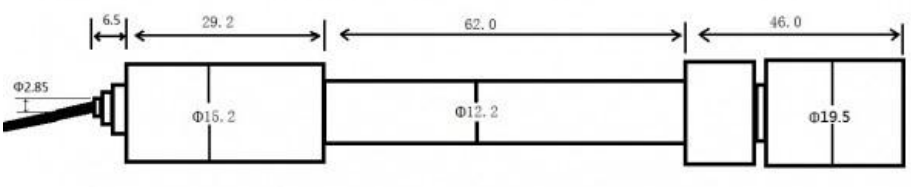
Applications

- Water quality testing
- Aquaculture

Specification

- Module Power：5.00V Module
- Size：43mm×32mm Measuring
- Range:0-14PH Measuring
- Temperature :0-60 ℃ Accuracy：
- ± 0.1pH (25 ℃) Response Time：
- ≤ 1min
- pH Sensor with BNC Connector
- PH2.0 Interface (3 foot patch)
- Gain Adjustment Potentiometer
- Power Indicator LED
- Cable Length from sensor to BNC connector:660mm

pH Electrode Size



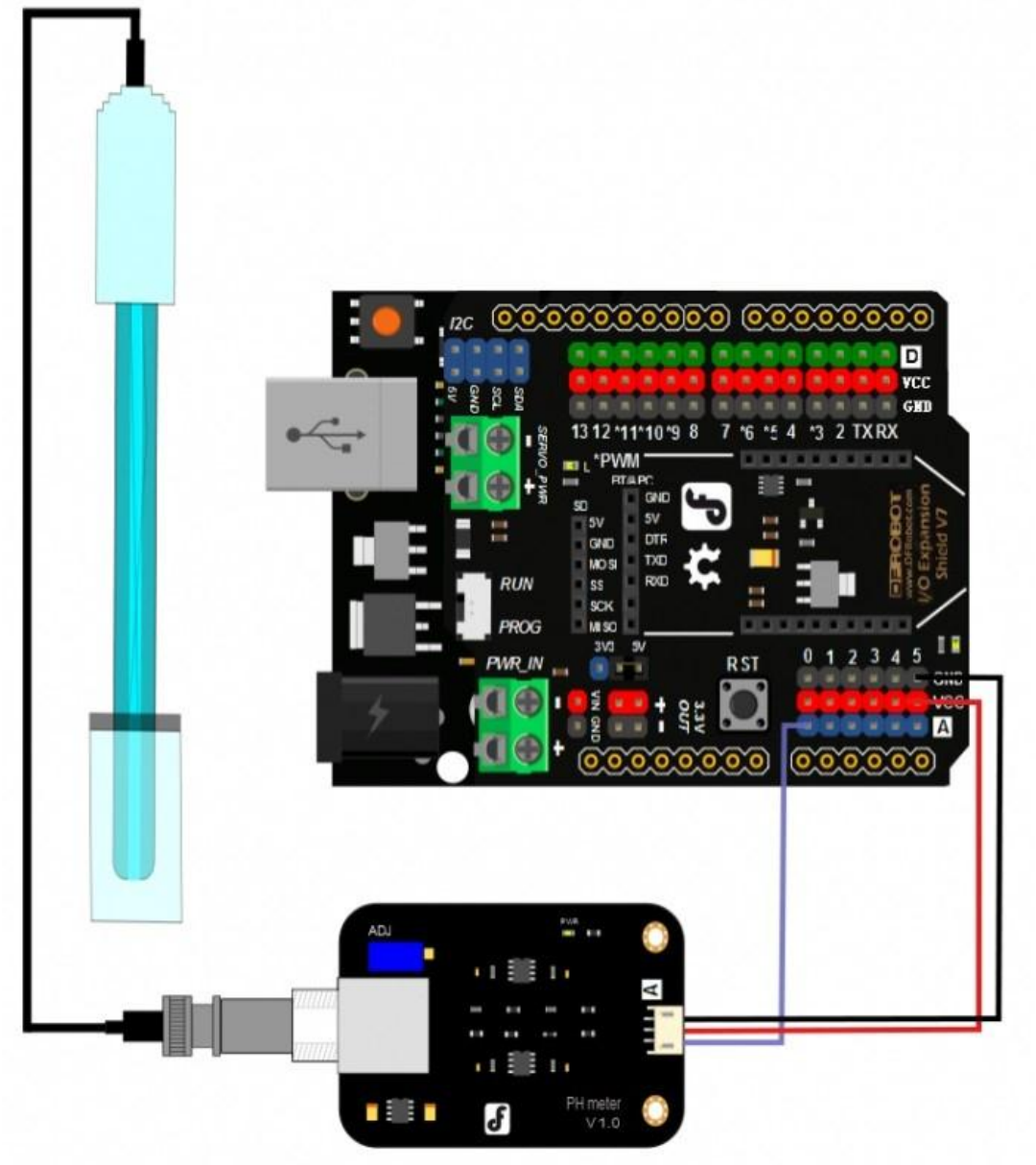
pH Electrode Characteristics

The output of pH electrode is Millivolts,and the pH value of the relationship is shown as follows (25 ℃):

VOLTAGE (mV)	pH value	VOLTAGE (mV)	pH value
414.12	0.00	-414.12	14.00
354.96	1.00	-354.96	13.00
295.80	2.00	-295.80	12.00
236.64	3.00	-236.64	11.00
177.48	4.00	-177.48	10.00
118.32	5.00	-118.32	9.00
59.16	6.00	-59.16	8.00
0.00	7.00	0.00	7.00

Use the pH Meter

Connecting Diagram



Step to Use the pH Meter

Cautions:

- Please use an external switching power supply,and the voltage as close as possible to the +5.00V. More accurate the voltage, more higher the accuracy!
- Before the electrode in continuous use every time,you need to calibrate it by the standard solution,in order to obtain more accurate results.The best environment temperature is about 25 ℃,and the pH value is known and reliable,close to the measured value. If you measure the acidic sample, the pH value of the standard solution should

be 4.00.If you measure the alkaline sample, the pH value of the standard solution should be 9.18.Subsection calibration, just in order to get a better accuracy.

- Before the pH electrode measured different solutions, we need to use water to wash it. We recommend using deionized water.

(1) Connect equipments according to the graphic,that is,the pH electrode is connected to the BNC connector on the pH meter board and then use the connection lines,the pH meter board is connected to the ananlong port 0 of the Arduino controller. When the Arduino controller gets power,you will see the blue LED on board is on.
(2) Upload the sample code to the Arduino controller.
(3) Put the pH electrode into the standard solution whose pH value is 7.00 or directly shorted the input of the BNC connector.Open the serial monitor of the Arduino IDE,you can see the pH value printed on it,and the error does not exceed 0.3. Record the pH value printed,then compared with 7.00, and the difference should be changed into the "Offset" in the sample code. For example,the pH value printed is 6.88,so the difference is 0.12.You should change the "# define Offset 0.00" into "# define Offset 0.12" in your program.
(4) Put the pH electrode into the pH standard solution whose value is 4.00.Then wait about one minute,adjust the gain potential device, let the value stabilise at around 4.00.At this time,the acidic calibration has been completed and you can measure the pH value of an acidic solution.
Note:If you want to measure the pH value of other solution,you must wash the pH electrode first!
(5) According to the linear characteristics of pH electrode itself, after the above calibration,you can directly measure the pH value of the alkaline solution, but if you want to get better accuracy, you can recalibrate it. Alkaline calibration use the standard solution whose pH value is 9.18.Also adjust the gain potential device, let the value stabilise at around 9.18. After this calibration, you can measure the pH value of the alkaline solution.

Sample Code

Sample code for testing the PH meter and get the sensor feedback from the Arduino Serial Monitor.

```
/*
 # This sample code is used to test the pH meter V1.0.
 # Editor : YouYou
 # Ver   : 1.0
 # Product: analog pH meter
 # SKU   : SEN0161
*/
#define SensorPin A0          //pH meter Analog output to Arduino Analog Input 0
#define Offset 0.00           //deviation compensate
#define LED 13
#define samplingInterval 20
#define printInterval 800
#define ArrayLenth 40    //times of collection
int pHArray[ArrayLenth];   //Store the average value of the sensor feedback
int pHArrayIndex=0;
void setup(void)
{
  pinMode(LED,OUTPUT);
  Serial.begin(9600);
  Serial.println("pH meter experiment!");    //Test the serial monitor
}
void loop(void)
{
  static unsigned long samplingTime = millis();
  static unsigned long printTime = millis();
  static float pHValue,voltage;
  if(millis()-samplingTime > samplingInterval)
  {
      pHArray[pHArrayIndex++]=analogRead(SensorPin);
      if(pHArrayIndex==ArrayLenth)pHArrayIndex=0;
      voltage = avergearray(pHArray, ArrayLenth)*5.0/1024;
      pHValue = 3.5*voltage+Offset;
      samplingTime=millis();
  }
  if(millis() - printTime > printInterval)   //Every 800 milliseconds, print a numerical, convert the state of the LED indicator
  {
 Serial.print("Voltage:");
        Serial.print(voltage,2);
        Serial.print("    pH value: ");
 Serial.println(pHValue,2);
        digitalWrite(LED,digitalRead(LED)^1);
        printTime=millis();
  }
}
```

```
double avergearray(int* arr, int number){
  int i;
```

```
int max,min;
double avg;
long amount=0;
if(number<=0){
 Serial.println("Error number for the array to avraging!/n");
 return 0;
}
if(number<5){   //less than 5, calculated directly statistics
 for(i=0;i<number;i++){
  amount+=arr[i];
 }
 avg = amount/number;
 return avg;
}else{
 if(arr[0]<arr[1]){
  min = arr[0];max=arr[1];
 }
 else{
  min=arr[1];max=arr[0];
 }
 for(i=2;i<number;i++){
  if(arr[i]<min){
   amount+=min;       //arr<min
   min=arr[i];
  }else {
   if(arr[i]>max){
    amount+=max;    //arr>max
    max=arr[i];
   }else{
    amount+=arr[i]; //min<=arr<=max
   }
  }//if
 }//for
 avg = (double)amount/(number-2);
}//if
return avg;
}
```

Precautions

- The electrode used for the first or long set without re-use, the electrode bulb and the sand core, immersed in the 3NKCL solution activated eight hours.
- The electrode plug should be kept clean and dry.
- Electrode reference solution is the 3NKCL solution.
- Measurement should be avoided staggered pollution between solutions, so as not to affect the accuracy of measurement.
- Electrode blub or sand core is defiled which will make PTS decline, slow response. So, it should be based on the characteristics of the pollutant, adapted to the cleaning solution, the electrode performance recovery.
- The electrode should not be long-term immersed in acid chloride solution.
- Electrode when in use, the ceramic sand core and liquid outlet rubber ring should be removed, in order to make salt bridge solution to maintain a certain velocity.

Documents

Schematic
PCB Design layout
pH Electrode Manual
Arduino Sample Code
Zips For All Above

→ Go Shopping pH meter(SKU:SEN0161)

Category: DFRobot > Sensors & Modules > Sensors > Liquid Sensors

What links here

Related changes

Special pages

Permanent link

Page information

Ultrasonic Ranging Module HC - SR04

Product features:

Ultrasonic ranging module HC - SR04 provides 2cm - 400cm non-contact measurement function, the ranging accuracy can reach to 3mm. The modules includes ultrasonic transmitters, receiver and control circuit. The basic principle of work:
(1) Using IO trigger for at least 10us high level signal,
(2) The Module automatically sends eight 40 kHz and detect whether there is a pulse signal back.
(3) IF the signal back, through high level , time of high output IO duration is the time from sending ultrasonic to returning.
Test distance = (high level time×velocity of sound (340M/S) / 2,

Wire connecting direct as following:

5V Supply
Trigger Pulse Input
Echo Pulse Output
0V Ground

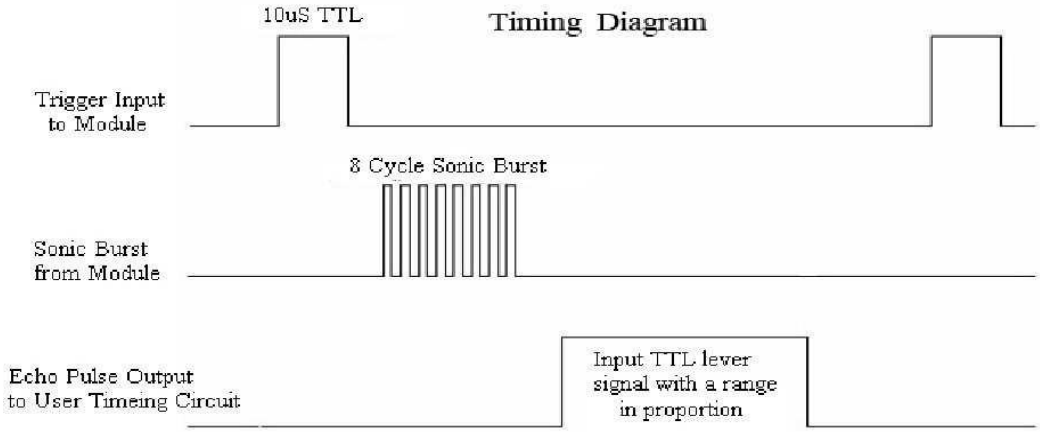
Electric Parameter

Working Voltage	**DC 5 V**
Working Current	**15mA**
Working Frequency	**40Hz**
Max Range	**4m**
Min Range	**2cm**
MeasuringAngle	**15 degree**
Trigger Input Signal	**10uS TTL pulse**
Echo Output Signal	**Input TTL lever signal and the range in proportion**
Dimension	**45*20*15mm**

Vcc Trig Echo GND

Timing diagram

The Timing diagram is shown below. You only need to supply a short 10uS pulse to the trigger input to start the ranging, and then the module will send out an 8 cycle burst of ultrasound at 40 kHz and raise its echo. The Echo is a distance object that is pulse width and the range in proportion .You can calculate the range through the time interval between sending trigger signal and receiving echo signal. Formula: uS / 58 = centimeters or uS / 148 =inch; or: the range = high level time * velocity (340M/S) / 2; we suggest to use over 60ms measurement cycle, in order to prevent trigger signal to the echo signal.



Attention:

- The module is not suggested to connect directly to electric, if connected electric, the GND terminal should be connected the module first, otherwise, it will affect the normal work of the module.
- When tested objects, the range of area is not less than 0.5 square meters and the plane requests as smooth as possible, otherwise ,it will affect the results of measuring.

Mouser Electronics

Authorized Distributor

Click to View Pricing, Inventory, Delivery & Lifecycle Information:

SparkFun Electronics:
SEN-13959